A Joint Standard of AASHTO, ITE, and NEMA

NTCIP 1202:2005 v02.19

National Transportation Communications for ITS Protocol

Object Definitions for Actuated Traffic Signal Controller (ASC) Units – version 02

November 2005

A major revision of NTCIP 1202:1997 v01.07, and includes the draft 1202 Amendment 1 v06

This Adobe® PDF copy of an NTCIP standard is available at no-cost for a limited time through support from the U.S. DOT / Federal Highway Administration, whose assistance is gratefully acknowledged.

Published by

American Association of State Highway and Transportation Officials (AASHTO)
444 North Capitol Street, 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 1752
Rosslyn, Virginia 22209-3806

NOTICES

Copyright Notice

PDF File License Agreement

Data Dictionary and MIB Distribution Permission

(ii) the copies or derivative works are not made part of the standards publications or works offered by other standards developing organizations or publishers or as works-for-hire not associated with commercial hardware or software products intended for field implementation;
(iii) use of the DD or MIB is restricted in that the syntax fields may be modified only to reflect a more restrictive subrange or enumerated values;
(iv) the description field may be modified but only to the extent that: (a) only those bit values or enumerated values that are supported are listed; and (b) the more restrictive subrange is expressed.

These materials are delivered "AS IS" without any warranties as to their use or performance.

AASHTO / ITE / NEMA and their suppliers do not warrant the performance or results you may obtain by using these materials. AASHTO / ITE / NEMA and their suppliers make no warranties, express or implied, as to noninfringement of third party rights, merchantability, or fitness for any particular purpose. In no event will AASHTO / ITE / NEMA or their suppliers be liable to you or any third party for any claim or for any consequential, incidental or special damages, including any lost profits or lost savings, arising from your reproduction or use of these materials, even if an AASHTO / ITE / NEMA representative has been advised of the possibility of such damages.

Some states or jurisdictions do not allow the exclusion or limitation of incidental, consequential or special damages, or the exclusion of implied warranties, so the above limitations may not apply to you.

Use of these materials does not constitute an endorsement or affiliation by or between AASHTO, ITE, or NEMA and you, your company, or your products and services.

If you are unwilling to accept the foregoing restrictions, you should immediately return these materials.

PRL and RTM Distribution Permission

To the extent that these materials are distributed by AASHTO / ITE / NEMA in the form of a Profile Requirements List ("PRL") or a Requirements Traceability Matrix ("RTM"), AASHTO / ITE / NEMA extend the following permission:

(i) you may make and/or distribute unlimited copies, including derivative works of the PRL (then known as a Profile Implementation Conformance Statement ("PICS")) or the RTM, provided that each copy you make and/or distribute contains the citation "Based on NTCIP 0000 [insert the document number] PRL or RTM. Used by permission. Original text (C) AASHTO / ITE / NEMA.";
(ii) you may not modify the PRL or the RTM except for the Project Requirements column, which is the only column that may be modified to show a product's implementation or the project-specific requirements; and
(iii) if the PRL or RTM excerpt is made from an unapproved draft, add to the citation "PRL (or RTM) excerpted from a draft document containing preliminary information that is subject to change."

This limited permission does not include reuse in works offered by other standards developing organizations or publishers, and does not include reuse in works-for-hire, compendiums, or electronic storage devices that are not associated with commercial hardware or software products intended for field installation.

A PICS is a Profile Requirements List which is completed to indicate the features that are supported in an implementation. Visit www.ntcip.org for information on electronic copies of the MIBs, PRLs, and RTMs.

Content and Liability Disclaimer

The information in this publication was considered technically sound by the consensus of persons engaged in the development and approval of the document at the time it was developed. Consensus does not necessarily mean that there is unanimous agreement among every person participating in the development of this document.

AASHTO, ITE, and NEMA standards and guideline publications, of which the document contained herein is one, are developed through a voluntary consensus standards development process. This process brings together volunteers and/or seeks out the views of persons who have an interest in the topic covered by this publication. While AASHTO, ITE, and NEMA administer the process and establish rules to promote fairness in the development of consensus, they do not write the document and they do not independently test, evaluate, or verify the accuracy or completeness of any information or the soundness of any judgments contained in their standards and guideline publications.

AASHTO, ITE, and NEMA disclaim liability for any personal injury, property, or other damages of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, application, or reliance on this document. AASHTO, ITE, and NEMA disclaim and make no guaranty or warranty, express or implied, as to the accuracy or completeness of any information published herein, and disclaims and makes no warranty that the information in this document will fulfill any of your particular purposes or needs. AASHTO, ITE, and NEMA do not undertake to guarantee the performance of any individual manufacturer or seller's products or services by virtue of this standard or guide.

In publishing and making this document available, AASHTO, ITE, and NEMA are not undertaking to render professional or other services for or on behalf of any person or entity, nor are AASHTO, ITE, and NEMA undertaking to perform any duty owed by any person or entity to someone else. Anyone using this document should rely on his or her own independent judgment or, as appropriate, seek the advice of a competent professional in determining the exercise of reasonable care in any given circumstances. Information and other standards on the topic covered by this publication may be available from other sources, which the user may wish to consult for additional views or information not covered by this publication.

AASHTO, ITE, and NEMA have no power, nor do they undertake to police or enforce compliance with the contents of this document. AASHTO, ITE, and NEMA do not certify, test, or inspect products, designs, or installations for safety or health purposes. Any certification or other statement of compliance with any health or safety–related information in this document shall not be attributable to AASHTO, ITE, or NEMA and is solely the responsibility of the certifier or maker of the statement.

Trademark Notice

NTCIP is a trademark of AASHTO / ITE / NEMA. All other marks mentioned in this document are the trademarks of their respective owners.

ACKNOWLEDGEMENTS

This publication was prepared by the NTCIP Actuated Signal Controller Working Group, which is a subdivision of the Joint Committee on the NTCIP. The Joint Committee is organized under a Memorandum of Understanding among the American Association of State Highway and Transportation Officials (AASHTO), the Institute of Transportation Engineers (ITE), and the National Electrical Manufacturers Association (NEMA). The Joint Committee on the NTCIP consists of six representatives from each of the standards organizations, and provides guidance for NTCIP development.

At the time that this document was prepared, the following individuals were members of the NTCIP Actuated Signal Controller Working Group:

- Robert DeRoche
- Tod Eidson
- W. L. (Bud) Kent
- Bruce McEntire
- Gary Meredith
- Peter Ragsdale (Co-chair)
- Elizabeth Ramirez (Co-chair)
- Joerg (Nu) Rosenbohm

Other individuals providing input to the document include:

- Blake Christie
- Larry Lindley

In addition to the many volunteer efforts, recognition is also given to those organizations who supported the efforts of the working groups by providing comments and funding for the standard, including:

- City of Dallas
- City of Phoenix
- Siemens ITS Eagle Traffic Control Systems
- Econolite Control Products, Inc.
- Siemens ITS Gardner Transportation Systems
- Iteris, Inc.
- Kimley Horne and Associates
- McCain Traffic Supply, Inc.
- PB Farradyne, a Division of PBQD, Inc.
- Peek Traffic Corporation, Inc., a member of Quixote Traffic
- Robert DeRoche Consulting
- Trevilon Corp.
- U.S. Department of Transportation, Federal Highway Administration

FOREWORD

This document defines the Actuated Traffic Signal Controller Unit (ASC) objects that are supported by devices that are NTCIP-compliant. There are three normative annexes and one informative annex to this document.

The first version of this document, now called version 01, was published as NEMA TS 3.5-1996. In 1997, both AASHTO and ITE balloted and approved the standard. This version 02 was developed to reflect lessons learned, to update the document to the latest documentation format, and to add new features.

This document is an NTCIP Device Data Dictionary standard. Device Data Dictionary standards provide definitions of data elements for use within NTCIP systems. A Joint NTCIP Device Data Dictionary standards publication is equivalent to these document types at the standards organizations:

AASHTO – Standard Specification
ITE – Software Standard
NEMA – Standard

For more information about NTCIP standards, visit the NTCIP Web Site at http://www.ntcip.org. For a hardcopy summary of NTCIP information, contact the NTCIP Coordinator at the address below.

User Comment Instructions

The term “User Comment” includes any type of written inquiry, comment, question, or proposed revision, from an individual person or organization, about any part of this standard publication’s content. A “request for interpretation” of this standard publication is also classified as a User Comment. User Comments are solicited at any time. In preparation of this NTCIP standards publication, input of users and other interested parties was sought and evaluated.

All User Comments will be referred to the committee responsible for developing and/or maintaining this standards publication. The committee chairperson, or their designee, may contact the submitter for clarification of the User Comment. When the committee chairperson or designee reports the committee’s consensus opinion related to the User Comment, that opinion will be forwarded to the submitter. The committee chairperson may report that action on the User Comment may be deferred to a future committee meeting and/or a future revision of the standards publication. Previous User Comments and their disposition may be available for reference and information at www.ntcip.org.

A User Comment should be submitted to this address:

NTCIP Coordinator
National Electrical Manufacturers Association
1300 North 17th Street, Suite 1752
Rosslyn, Virginia 22209-3806
fax: (703) 841-3331
e-mail: ntcip@nema.org

A User Comment should be submitted in the following form:

Standard Publication number and version:
Page:
Paragraph or Clause:
Comment:

Please include your name, organization, and address in your correspondence.

Approvals

This document was separately balloted and approved by AASHTO, ITE, and NEMA after recommendation by the Joint Committee on the NTCIP. Each organization has approved this standard as the following standard type, as of the date:

AASHTO – Standard Specification; November 2004
ITE – Software Standard; March 2005
NEMA – Standard; November 2004

History

From 1996 to 1999, this document was referenced as NEMA TS 3.5-1996. However, to provide an organized numbering scheme for the NTCIP documents, this document is now referenced as NTCIP 1202. The technical specifications of NTCIP 1202 are identical to the former reference, except as noted in the development history below, and in the following Index of Revisions.

NEMA TS 3.5-1996. 1996 – Approved by NEMA. 1996 – Accepted as a Recommended Standard by the Joint Committee on the NTCIP. 1997 – Approved by AASHTO and ITE. v01.07a printed with NEMA cover.

NTCIP 1202 v01. v01.07b printed with joint cover. v01.07c printed to PDF in November 2002. v01.07d printed to PDF for no-cost distribution January 2005.

NTCIP 1202 Amendment 1. November 1999 – Accepted as a User Comment Draft Amendment by the Joint Committee on the NTCIP. April 2000 – NTCIP Standards Bulletin B0049 sent NTCIP 1202 Amendment 1 v01.06b for user comment. NTCIP 1202 Amendment 1, a User Comment Draft, was incorporated into 1202v02, and was not advanced further.

NTCIP 1202 v02.10. June 2001 – Accepted as a User Comment Draft by the Joint Committee on the NTCIP. February 2002 – NTCIP Standards Bulletin B0068 referred v02.13 for user review and comment.

NTCIP 1202 v02.16. October 2002 – Accepted as a Recommended Standard by the Joint Committee on the NTCIP. April 2004 – NTCIP Standards Bulletin B0091 referred v02.18 for balloting. Approved by AASHTO in November 2004, approved by ITE in March 2005, and approved by NEMA in November 2004.

NTCIP 1202:2005 v02.19. November 2005 – Edited document for publication. By the terms of MOU on CTPA article 1.2, the ownership of version 02 was assigned to AASHTO, ITE, and NEMA because the preexisting work was revised by more than 50%.

Compatibility of Versions

All NTCIP Standards Publications have a major and minor version number for configuration management. The version number syntax is "v00.00a," with the major version number before the period, and the minor version number and edition letter (if any) after the period.

Anyone using this document should seek information about the version number that is of interest to them in any given circumstance. The MIB, the PRL, and the PICS should all reference the version number of the standards publication that was the source of the excerpted material.

Compliant systems based on later, or higher, version numbers MAY NOT be compatible with compliant systems based on earlier, or lower, version numbers. Anyone using this document should also consult NTCIP 8004 for specific guidelines on compatibility.

INTRODUCTION

This publication defines data elements for use with Actuated Signal Controller Units. The data is defined using the Simple Network Management Protocol (SNMP) object-type format as defined in RFC 1212 and the defined NTCIP format defined in NTCIP 8004. This data would typically be exchanged using one of the NTCIP 1103 recognized Application Layers (e.g., SNMP).

This standard defines requirements that are applicable to all NTCIP environments and it also contains optional and conditional clauses that are applicable to specific environments for which they are intended.

The following keywords apply to this document: AASHTO, ITE, NEMA, NTCIP, data, data dictionary, object.

In 1992, the NEMA 3-TS Transportation Management Systems and Associated Control Devices Section began the effort to develop the NTCIP. Under the guidance of the Federal Highway Administration’s NTCIP Steering Group, the NEMA effort was expanded to include the development of communications standards for all transportation field devices that could be used in an ITS network.

In September 1996, an agreement was executed among AASHTO, ITE, and NEMA to jointly develop, approve, and maintain the NTCIP standards. In late 1998, the Actuated Signal Controller Working Group was tasked with the effort to update the Actuated Traffic Signal Controller Object Definitions document. The first meeting of this working group was held in October 1999.

CONTENTS

INDEX OF REVISIONS IN v02.10

The following is a list by page and/or section of the revisions included in the v02.10 draft of this Standard:

ALL SECTIONS:
Changed: references from TS 3.4 to NTCIP 1201

SECTION 2:
All Objects - Status from Mandatory to Optional – Changed
Page 18 - 2.2.2.20 Phase Startup - Changed
Page 33 - 2.3.2.2 Vehicle Detector Options Parameter - Changed
Page 48 - 2.4.3 Backup Time Parameter - Changed
Page 49 - 2.4.5 Unit control Status - Changed
Page 50 - 2.4.6 Unit Flash Status - Changed
Page 50 - 2.4.7 Unit Alarm Status 2 - Changed
Page 52 - 2.4.9 Short Alarm Status - Changed
Page 52 - 2.4.10 Unit Control - Changed
Page 56 - 2.4.14.2 Special Function Output State - Changed
Page 56 - 2.4.14.3 Special Function Output Control - Changed
Page 56 - 2.4.14.4 Special Function Output Status - Changed
Page 59 - 2.5.6 Pattern Table Type - Changed
Page 62 - 2.5.7.4 Pattern Split Number Parameter - Changed
Page 68 - 2.5.14 System Pattern Control - Changed
Page 69 - 2.5.15 System Sync Control - Changed
Page 71 - 2.6.3.2 Time Base Action Pattern Parameter - Changed
Page 71 - 2.6.3.3 Time Base Action Auxiliary Function Parameter - Changed
Page 74 - 2.7.2.2 Preempt Control Parameter - Changed
Page 80 - 2.7.2.16 Preempt Status – Changed
Page 81 - 2.7.2.17 Preempt Track Overlap Parameter - Added
Page 81 - 2.7.2.18 Preempt Dwell Overlap Parameter - Added
Page 85 - 2.7.3.2 Preempt Control State - Changed
Page 89 - 2.8.5.2 Ring Stop Time Control - Changed
Page 89 - 2.8.5.3 Ring Force Off Control - Changed
Page 90 - 2.8.5.4 Ring Max 2 Control - Changed
Page 90 - 2.8.5.5 Ring Max Inhibit Control - Changed
Page 91 - 2.8.5.6 Ring Ped Recycle Control - Changed
Page 91 - 2.8.5.7 Ring Red Rest Control - Changed
Page 92 - 2.8.5.8 Ring Omit Red Control - Changed
Page 96 - 2.9.2.4 Channel Flash Parameters - Changed
Page 96 - 2.9.2.5 Channel Dim Parameters - Changed
Page 109 - 2.12.ASC Block Objects - Added
Page 109 - 2.12.1 ASC Block Get Control - Added
Page 110 - 2.12.2 ASC Block Data - Added

SECTION 3:
Page 113- Section 3 for Block Object Definitions - Added

ANNEX A:
Page 139 - A.7 Special Function Conformance Group - Changed
Page 145 - A.16 Database Management Conformance Group (Object Status) - Changed
Page 145 - A.19 Report Conformance Group - Changed
Page 144 - A.20 Block Object Conformance Group - Added

ANNEX B:
Page 155 - Expanded definitions along with examples - Added

INDEX OF REVISIONS IN v02.11

The following is a list by page and/or section of the revisions included in the v02.11 draft of this Standard:

SECTION 2:
Page 27 - 2.2.5 Phase Control Table - Changed
Page 52 - 2.4.10 Unit Control - Changed
Page 62 - 2.5.7.5 Pattern Sequence Number - Changed
Page 67 - 2.5.11 Local Free Status - Changed
Page 71 - 2.6.3.3 Time Base Action Auxiliary Function - Changed
Page 74 - 2.7.2.2 Preempt Control - Changed
Page 96 - 2.9.2.4 Channel Flash - Changed
Page 96 - 2.9.2.5 Channel Dim - Changed
Page 109 - 2.12.1 ASC Block Get Control
Page 110 - 2.12.2 ASC Block Data

SECTION 3:
Section 3 for Block Object Definitions - Changed

ANNEX A:
Page 143 - A.13 Overlap Conformance Group - Changed

ANNEX B:
Page 155 - B.1 Consistency Check Rules - Changed

ANNEX C:
Page 159 - Annex C - Added

INDEX OF REVISIONS IN v02.12

The following is a list by page and/or section of the revisions included in the v02.12 draft of this Standard:

GENERAL:
Cover Sheet & Following Page - Changed
Page xii - Acknowledgements - Added
Page ii - Foreword - Changed
Page xii - Introduction - Added
Page xiii - v02.08 Revisions - Deleted
Page xiii - v02.09 Revisions - Deleted
Page xiv - v02.12 Revisions - Added

SECTION 1:
Page 1 - 1.1 Scope - Changed
Page 1 - 1.2 References - Changed
Page 1 - 1.2.1 Normative References - Changed
Page 2 - 1.22 Other References - Changed
Page 3 - 1.3 Actuated Controller Unit Terms - Changed

SECTION 2:
Page All – Meta Commands - Added
Section 2 Object Definitions - Added
Page 50 - 2.4.7 Unit Alarm Status 2 - Changed
Page 81 - 2.7.2.19 Preempt Cycling Phase Parameters - Added
Page 82 - 2.7.2.20 Preempt Cycling Ped Parameters - Added
Page 82 - 2.7.2.21 Preempt Cycling Overlap Parameters - Added
Page 92 - 2.8.6 Ring Status Table - Added

Page 93 - 2.8.6.1 Ring Status - Added
Page 101 - 2.10.2.2 Overlap Type - Changed
Page 102 - 2.10.2.4 Overlap Modifier Phase Parameters - Changed
Page 109 - 2.12.1 Block Get Control - Changed
Page 110 - 2.12.2 Block Data – Changed
Page 111 - 2.12.3 Block Error Status - Added

SECTION 3:
Page Numerous - Block Examples - Added
Page 118 - 3.8 Block Preempt Data - Changed
Page 129 - 3.18 Block Dynamic Object Owner Data - Changed
Page 130 - 3.19 Block Dynamic Object Status Data - Changed

ANNEX A:
Page 146 - A.19 Aux IO Group - Added
Page 146 - A.20 PMPP Group - Added
Page 146 - A.21 SNMP Group - Added
Page 147 - A.22 System Group - Added
Page 148 - A.23 SFMP Group - Added
Page 148 - A.24 STMP Group - Added
Page 149 - A.25 Logical Name Group - Added
Page 149 - A.26 Trap Management Group - Added
Page 150 - A.27 Security Group - Added
Page 150 - A.28 RS232 Group - Added
Page 151 - A.29 HDLC Group - Added
Page 152 - A.30 Interfaces Group - Added
Page 152 - A.31 IP Group - Added
Page 154 - A.32 ICMP Group - Added
Page 154 - A.33 TCP Group - Added
Page 155 - A.34 UDP Group - Added
Page 155 - A.35 Ethernet Group

ANNEX B:
Page 155 - B.1 Consistency Check Rules - Changed

INDEX OF REVISIONS IN v02.13

The following is a list by page and/or section of the revisions included in the v02.13 User Comment Draft of this Standard:

GENERAL:
Title Page & Following Page – Revised
Page xii – Acknowledgements – Revised
Page ii – Foreword – Revised; added Approvals and History
Page xii – Introduction - Revised
Page xv – v02.13 Revisions – Added

INDEX OF REVISIONS IN v02.14

The following is a list by page and/or section of the revisions included in the v02.14 draft of this Standard:

GENERAL:

Page xvi – v02.14 Revisions - Added

SECTION 2:

Page 28 - 2.2.5.2 Phase Omit Control - Changed
Page 29 - 2.2.5.3 Pedestrian Omit Control - Changed
Page 29 - 2.2.5.4 Phase Hold Control - Changed
Page 30 - 2.2.5.5 Phase Force Off Control - Changed
Page 30 - 2.2.5.6 Vehicle Call Control - Changed
Page 31 - 2.2.5.7 Pedestrian Call Control - Changed
Page 33- 2.3.2.2 Vehicle Detector Options Parameter - Changed
Page 43 - 2.3.5.4.1 Volume Data - Changed
Page 43 - 2.3.5.4.2 Occupancy data - Changed
Page 48 - 2.4.3 Backup Time Parameter
Page 52 - 2.4.10 Unit Control - Changed
Page 56 - 2.4.14.3 Special Function Output Control - Changed
Page 68 - 2.5.14 System Pattern Control - Changed
Page 69 - 2.5.15 System Sync Control - Changed
Page 69 - 2.6.1 Time Base Pattern Sync Parameter - Changed
Page 62 - 2.5.7.3 Pattern Offset Time Parameter - Changed
Page 71 - 2.6.3.1 Time Base Action Number - Changed
Page 71 - 2.6.3.2 Time Base Action Pattern Parameter - Changed
Page 73 - 2.7.2 Preempt Table - Changed
Page 74 - 2.7.2.1 Preempt Number - Changed
Page 74 - 2.7.2.2 Preempt Control Parameter - Changed
Page 75 - 2.7.2.3 Preempt Link Parameter - Changed
Page 76 - 2.7.2.4 Preempt Delay Parameter - Added DEFVAL
Page 76 - 2.7.2.5 Preempt Duration Parameter - Changed
Page 76 - 2.7.2.6 Preempt Minimum Green Parameter - Changed
Page 77 - 2.7.2.7 Preempt Minimum Walk Parameter - Changed
Page 77 - 2.7.2.8 Preempt Enter Ped Clear Parameter - Changed
Page 77 - 2.7.2.9 Preempt Track Green Parameter - Changed
Page 78 - 2.7.2.10 Preempt Minimum Dwell Parameter - Changed
Page 78 - 2.7.2.11 Preempt Maximum Presence Parameter - Added DEFVAL
Page 79 - 2.7.2.12 Preempt Track Phase Parameter - Changed
Page 79 - 2.7.2.13 Preempt Dwell Phase Parameter - Changed
Page 79 - 2.7.2.14 Preempt Dwell Ped Parameter - Changed
Page 80 - 2.7.2.15 Preempt Exit Phase Parameter - Changed
Page 81 - 2.7.2.17 Preempt Track Overlap Parameter - Changed
Page 81 - 2.7.2.18 Preempt Dwell Overlap Parameter - Changed
Page 81 - 2.7.2.19 Preempt Cycling Phase Parameter - Changed
Page 82 - 2.7.2.20 Preempt Cycling Ped Parameter - Changed
Page 82 - 2.7.2.21 Preempt Cycling Overlap Parameter - Changed
Page 82 - 2.7.2.22 Preempt Enter Yellow Change Parameter - Added
Page 83 - 2.7.2.23 Preempt Enter Red Clear Parameter - Added
Page 83 - 2.7.2.24 Preempt Track Yellow Change Parameter - Added
Page 83 - 2.7.2.25 Preempt Track Red Clear Parameter - Added
Page 85 - 2.7.3.2 Preempt Control State - Changed
Page 87 - 2.8.3.3 Sequence Data

Page 89 - 2.8.5.2 Ring Stop Time Control - Changed
Page 89 - 2.8.5.3 Ring Force Off Control - Changed
Page 90 - 2.8.5.4 Ring Max 2 Control - Changed
Page 90 - 2.8.5.5 Ring Max Inhibit Control - Changed
Page 91 - 2.8.5.6 Ring Ped Recycle Control - Changed
Page 91 - 2.8.5.7 Ring Red Rest Control - Changed
Page 92 - 2.8.5.8 Ring Omit Red Control - Changed
Page 106 - 2.11 TS2 Port 1 Parameters - Changed
Page 106 - 2.11.2 Port 1 Table - Changed
Page 107 - 2.11.2.2 Port 1 Device Present - Changed

SECTION 3:
Page 113 - 3.2 Phase Block Data - Changed Heading
Page 114 - 3.3 Vehicle Detector Block Data - Changed Heading
Page 115 - 3.4 Pedestrian Detector Block Data - Changed Heading
Page 116 - 3.5 Pattern Block Data - Changed Heading
Page 117 - 3.6 Split Block Data - Changed Heading
Page 118 - 3.7 Time Base Block Data - Changed Heading
Page 117 - 3.8 Block Preempt Data - Changed
Page 119 - 3.8.1 Preempt Block Example
Page 120 - 3.9 Sequence Block Data - Changed Heading
Page 121 - 3.10 Channel Block Data - Changed Heading
Page 122 - 3.11 Overlap Block Data - Changed Heading
Page 123 - 3.12 Port 1 Block Data - Changed Heading
Page 123 - 3.13 Schedule Block Data - Changed Heading
Page 124 - 3.14 Day Plan Block Data - Changed Heading
Page 125 - 3.15 Event Log Config Block Data - Changed Heading
Page 127 - 3.16 Event Class Block Data - Changed Heading
Page 128 - 3.17 Dynamic Object Config Block Data - Changed Heading
Page 129 - 3.18 Dynamic Object Owner Block Data - Changed Heading
Page 130 - 3.19 Dynamic Object Status Block Data - Changed Heading
Page 131 - 3.20 Miscellaneous ASC Block Data - Changed Heading

ANNEX A:
Page 141 - A.10 Preempt Conformance Group - Changed

INDEX OF REVISIONS IN v02.15

The following is a list by page and/or section of the revisions included in the v02.15 draft of this Standard:

GENERAL:
Page xvii – v02.15 Revisions - Added

SECTION 2:
Page 9 - 2.2.1 Maximum Phases - Changed
Page 31 - 2.3.1 Maximum Vehicle Detectors - Changed
Page 39 - 2.3.3 Maximum Vehicle Detector Status Groups - Changed
Page 44 - 2.3.6 Maximum Pedestrian Detectors - Changed
Page 53 - 2.4.11 Maximum Alarm Groups - Changed
Page 55 - 2.4.13 Maximum Special Function Outputs - Changed
Page 51 - 2.4.8 Unit Alarm Status 1 - Changed
Page 52 - 2.4.9 Short Alarm Status - Changed
Page 59 - 2.5.5 Maximum Patterns - Changed
Page 61 - 2.5.7.2 Pattern Cycle time - Changed
page 63 - 2.5.8 Maximum Splits - Changed

Page 67 - 2.5.11 Local Free Status - Changed
Page 70 - 2.6.2 Maximum Time Base Actions - Changed
Page 73 - 2.7.1 Maximum Preempts - Changed
Page 85 - 2.8.1 Maximum Rings - Changed
Page 85 - 2.8.2 Maximum Sequences - Changed
Page 87 - 2.8.4 Maximum Ring Control Groups - Changed
Page 94 - 2.9.1 Maximum Channels - Changed
Page 97 - 2.9.3 Maximum Channel Status Groups - Changed
Page 99 - 2.10.1 Maximum Overlaps - Changed
Page 103 - 2.10.3 Maximum Overlap Status Groups - Changed
Page 106 - 2.11.1 Maximum Port 1 Addresses - Changed

ANNEX D:
Page 163 - D.1 Special Function Output State - Added

INDEX OF REVISIONS IN v02.16

The following is a list by page and/or section of the revisions included in the v02.16 accepted Recommended Standard of this Standard:

PAGES:
Page xviii – v02.16 Revisions – Added
Page Numerous – Updated meta attribute fields
Page 9 – 2.2.1 Maximum Phases – Changed
Page 67 – 2.5.12 Coordination Cycle Status - Changed
Page 68 – 2.5.13 Coordination Sync Status - Changed
Page 134 – A.1.1 Type Symbols - Changed

INDEX OF REVISIONS IN v02.17

The following is a list by page and/or section of the revisions included in the v02.17 draft of this Standard:

PAGES:
Page xviii – v02.17 Revisions – Added
Page 38 –2.3.2.14 Vehicle Detector RESET – Changed
Page 54 –2.4.12 Alarm Group TABLE – Typographical error
Page 69 –2.6.1 Time Base Pattern Sync PARAMETER – Changed
Page 71 –2.6.3.2 Time Base Action Pattern PARAMETER – Changed
Page 74 –2.7.2.2 Preempt Control PARAMETER – Changed
Page 85 –2.7.3.2 Preempt Control STATE – Changed
Page 107 – 2.11.2.2 Port 1 Device PRESENT – Changed
Page 108 – 2.22.2.3 Port 1 Frame 40 ENABLE – Changed
Page 134 – A.1.2 STATUS SYMBOLS – Typographical Error
Page 135 – A.2 ASC Requirements – Changed
Page 144 – A.16 Configuration Conformance Group – Changed
Page 146 – A.19 AuxIO Group – Changed
Page 146 – A.21 SNMP GROUP – Changed
Page 147 – A.22 SYSTEM GROUP – Changed
Page 150 – A.27 SECURITY GROUP – Changed
Page 155 – Annex B Consistency Checks – Changed
Page 155 – B.1 Consistency Check Rules – Changed
Page 155 – Annex C Concept of Operations – Heading changed
Page 162 – C.6 Overlap Supplemental – Changed

Page 163 – D.1 Special Function Output State – Changed

INDEX OF REVISIONS IN v02.18

The following is a list by page and/or section of the revisions included in the v02.18 ballot copy of this Standard:

GENERAL:

Formatted Headers and Footers
Added blank page note to even pages before new section

PAGES:

Page 55 – 2.4.14 Commented out specialFunctionOutputState in SpecialFunctionOutputEntry because it is deprecated.

INDEX OF REVISIONS IN v02.19

The following is a list by page and/or section of the revisions included in the v02.19 Jointly Approved and published edition of this Standard:

GENERAL:

Revised version and Joint Approval year in Headers and copyright year in Footers
Reorganized and edited front matter to comply with NTCIP 8002 A1 v02
Updated NEMA suite number

PAGES:

Page 2 – Section 1.2.1 References – updated IAB and RFC URLs.
Page 101 – 2.10.2.2 Changed overlapType ACCESS to read-write
Page 134 – A.2 Ref A.19 auxIO; corrected Clause reference to 1201 – 2.8
Page 141 – A.11 1202 Clause 2.8.5.1; changed Allowed Values to 1-255
Page 148 – A.26 Trap Management Group; deleted objects
Page 151 – A.30 Interface if.2.7; changed Allowed Values to 1-3
Page 151 – A.30 Interface if.2.8; changed Allowed Values to 1-3
Page 151 – A.31 IP ip.1; changed Allowed Values to 1-2
Page 152 – A.31 IP Group; changed one Object Type and four Allowed Values
Page 153 – A.33 TCP Group; changed five Allowed Values
Page 154 – A.34 UDP Group; changed four Allowed Values

< This page is intentionally left blank. >

Section 1
GENERAL

1.1 SCOPE

The messaging between Transportation Management and Actuated Signal Controllers is accomplished by using the NTCIP Application Layer services to convey requests to access or modify values stored in a given device; these values are referred to as objects.

The purpose of this publication is to identify and define those objects definitions that may be supported by an Actuated Signal Controller.

1.2 REFERENCES

For approved revisions, contact:

NTCIP Coordinator
National Electrical Manufacturers Association
1300 North 17th Street, Suite 1752
Rosslyn, VA 22209-3806
e-mail: ntcip@nema.org

For draft revisions, which are under discussion by the relevant NTCIP Working Group, and recommended revisions of the NTCIP Joint Committee, visit the world wide web at http://www.ntcip.org.

The following standards (normative references) contain provisions which, through reference in this text, constitute provisions of this Standard. Other documents and standards (other references) are referenced in these documents, which might provide a complete understanding of the entire protocol and the relations between all parts of the protocol. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this Standard are encouraged to investigate the possibility of applying the most recent editions of the standard listed below.

1.2.1 Normative References

ANSI
11 West 42nd Street, 13th Floor
New York, NY 10036

ISO/IEC 8824-1:1998	*Information Technology—Abstract Syntax Notation One (ASN.1): Specification of Basic Notation*

Internet Architecture Board
http://www.rfc-editor.org/
http://www.ietf.org/rfc.html
http://www.rfc-editor.org/repositories.html

Obtain Request for Comment (RFC) electronic documents from several repositories on the WWW, or by “anonymous” File Transfer Protocol (FTP) with several hosts. Browse or FTP to:

http://www.ietf.org/rfc.html
http://www.rfc-editor.org/
http://www.rfc-editor.org/repositories.html
for FTP sites, read ftp://ftp.isi.edu/in-notes/rfc-retrieval.txt

IAB STD 16	RFC 1155*: Structure and Identification of Management Information for TCP/IP based Internets,* May 1990 RFC 1212: *Concise MIB Definitions*, March 1991

National Electrical Manufacturers Association
1300 North 17th Street, Suite 1752
Rosslyn, VA 22209

NTCIP 1201	*NTCIP Global Object Definitions* (formerly numbered TS 3.4)

1.2.2 Other References

National Electrical Manufacturers Association
1300 North 17th Street, Suite 1752
Rosslyn, VA 22209

TS 2-1998	Traffic Controller Assemblies *with NTCIP Requirements*
NTCIP 1102	*NTCIP Octet Encoding Rules (OER) Base Protocol*
NTCIP 1103	*NTCIP Transportation Management Protocol*
NTCIP 9001	*The NTCIP Guide*

ANSI
11 West 42nd Street, 13th Floor
New York, NY 10036
(212) 642-4900

ISO/IEC 8825-1:1998	*Information Technology—ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).*

Internet Architecture Board
http://www.rfc-editor.org/
http://www.ietf.org/rfc.html
http://www.rfc-editor.org/repositories.html

Obtain Request for Comment (RFC) electronic documents from several repositories on the WWW, or by “anonymous” File Transfer Protocol (FTP) with several hosts. Browse or FTP to:

http://www.ietf.org/rfc.html
http://www.rfc-editor.org/
http://www.rfc-editor.org/repositories.html

for FTP sites, read ftp://ftp.isi.edu/in-notes/rfc-retrieval.txt

IAB STD 15 *RFC 1157: A Simple Network Management Protocol (SNMP),* May 1990

IAB STD 17 *RFC 1213: Management Information Base for Network Management of TCP/IP-based internets: MIB-II,* March 1991

1.3 ACTUATED CONTROLLER UNIT TERMS

These terms define the nomenclature frequently used in regard to actuated traffic signal control devices.

actuation The operation of any type of detector.

automatic flash Automatic programmed flash mode not caused by manual switch activation or fault condition or startup.

auxiliary function A control that may activate auxiliary functions or outputs in an actuated controller unit.

backup mode Control by local TBC or Interconnect based on absence of master or central command.

barrier A barrier (compatibility line) is a reference point in the preferred sequence of a multi-ring CU at which all rings are interlocked. Barriers assure there will be no concurrent selection and timing of conflicting phases for traffic movement in different rings. All rings cross the barrier simultaneously for the selection and timing of phases on the other side.

call A registration of a demand for right-of-way by traffic (vehicles or pedestrians) to a controller unit.

call, serviceable conflicting A call which:

a. Occurs on a conflicting phase not having the right-of-way at the time the call is placed.
b. Occurs on a conflicting phase which is capable of responding to a call.
c. When occurring on a conflicting phase operating in an occupancy mode, remains present until given its right-of-way.

channel Three circuits of a Monitor Device wired to monitor the green, yellow, and red outputs of the associated load switch position in the Terminal & Facilities. Channel 1 is assumed to monitor Load Switch 1, etc.

check An output from a controller unit that indicates the existence of unanswered call(s).

concurrency group A group of phases which describes possible timing combinations. A phase within the group shall be able to time concurrently with any other phase from another ring contained in the group. For example, in the typical dual-ring eight phase design, phases 1, 2, 5 and 6 form one concurrency group and phases 3, 4, 7, and 8 form another concurrency group.

concurrent timing A mode of controller unit operation whereby a traffic phase can be selected and timed simultaneously and independently with another traffic phase.

controller assembly A complete electrical device mounted in a cabinet for controlling the operation of a traffic control signal display(s).

controller unit A controller unit is that portion of a controller assembly that is devoted to the selection and timing of signal displays.

coordination The control of controller units in a manner to provide a relationship between specific green indications at adjacent intersections in accordance with a time schedule to permit continuous operation of groups of vehicles along the street at a planned speed.

coordinator A device or program/routine which provides coordination.

cycle The total time to complete one sequence of signalization around an intersection. In an actuated controller unit, a complete cycle is dependent on the presence of calls on all phases. In a pretimed controller unit it is a complete sequence of signal indications.

cycle length The time period in seconds required for one complete cycle.

detector, pedestrian A detector that is responsive to operation by or the presence of a pedestrian.

detector, system Any type of vehicle detector used to obtain representative traffic flow information.

detector, vehicle A detector that is responsive to operation by or the presence of a vehicle.

dial The cycle timing reference or coordination input activating same. Dial is also frequently used to describe the cycle.

display map A graphic display of the street system being controlled showing the status of the signal indications and the status of the traffic flow conditions.

dual entry Dual entry is a mode of operation (in a multi-ring CU) in which one phase in each ring must be in service. If a call does not exist in a ring when it crosses the barrier, a phase is selected in that ring to be activated by the CU in a predetermined manner.

dwell The interval portion of a phase when present timing requirements have been completed.

first coordinated phase The coordinated phase which occurs first within the concurrent group of phases containing the coordinated phase(s) when there are constant calls on all phases.

flash Operation where one section in each vehicle signal (yellow or red) is alternately on and off with a one second cycle time and a 50 percent duty cycle.

fault monitor state Internal CU diagnostics have determined that the CU device is not in a safe operational state. An output may be asserted to indicate this condition.

force off A command to force the termination of the green interval in the actuated mode or Walk Hold in the nonactuated mode of the associated phase. Termination is subject to the presence of a serviceable conflicting call. The Force Off function shall not be effective during the timing of the Initial, Walk, or Pedestrian Clearance. The Force Off shall only be effective as long as the condition is sustained. If a phase specific Force Off is applied, the Force Off shall not prevent the start of green for that phase.

Free Operation without coordination control from any source.

gap reduction A feature whereby the Unit Extension or allowed time spacing between successive vehicle actuations on the phase displaying the green in the extensible portion of the interval is reduced.

group Any portion of a traffic control network (system) that can be controlled by a common set of timing plans.

hold A command that retains the existing Green interval.

hold-on line A signal to an intersection controller commanding it to remain under computer control.

interconnect A means of remotely controlling some or all of the functions of a traffic signal.

intersection status The knowledge of whether a controlled intersection is on-line and which mode it is currently operating in.

interval The part or parts of the signal cycle during which signal indications do not change.

load switch driver group The set of three outputs which are used to drive load switch inputs to provide a Green, Yellow, or Red output condition for vehicle signals or Walk, Ped Clear, or Don’t Walk output condition for pedestrian signals.

malfunction management unit (MMU) A device used to detect and respond to improper and conflicting signals and improper operating voltages in a traffic controller assembly.

maximum green The maximum green time with a serviceable opposing actuation, which may start during the initial portion.

minimum green interval The shortest green time of a phase. If a time setting control is designated as Minimum Green, the green time shall be not less than that setting.

multi-ring controller unit A multi-ring CU contains two or more interlocked rings which are arranged to time in a preferred sequence and to allow concurrent timing of all rings, subject to barrier restraint.

nonlocking memory A mode of actuated-controller-unit operation which does not require the retention of a call for future utilization by the controller assembly.

occupancy A measurement of vehicle presence within a zone of detection, expressed in seconds of time a given point or area is occupied by a vehicle.

off-line A controller assembly not under the control of the normal control source.

offset The time relationship, expressed in seconds, between the starting point of the first coordinated phase Green and a system reference point. (See definition of First Coordinated Phase)

omit, phase A command that causes omission of a selected phase.

on-line A controller assembly under the control of the normal control source.

overlap A Green indication that allows traffic movement during the green intervals of and clearance intervals between two or more phases.

passage time The time allowed for a vehicle to travel at a selected speed from the detector to the stop line.

pattern A unique set of coordination parameters (cycle value, split values, offset value, and sequence).

pedestrian clearance interval The first clearance interval for the pedestrian signal following the pedestrian WALK indication.

pedestrian recycle A method of placing a recurring demand for pedestrian service on the movement when that movement is not in its Walk interval.

permissive A time period, during which the CU is allowed to leave the coordinated phase(s) under coordination control to go to other phases.

phase sequence A predetermined order in which the phases of a cycle occur.

phase, active The indicated phase is currently timing. A phase is always active if it is Green or Yellow (Walk or Pedestrian Clear for Pedestrian Phases). It is also active if it is timing Red Clearance. It may be considered active during Red Dwell.

phase, conflicting Conflicting phases are two or more traffic phases which will cause interfering traffic movements if operated concurrently.

phase, nonconflicting Nonconflicting phases are two or more traffic phases which will not cause interfering traffic movements if operated concurrently.

phase, pedestrian A traffic phase allocated to pedestrian traffic which may provide a right-of-way pedestrian indication either concurrently with one or more vehicular phases, or to the exclusion of all vehicular phases.

phase, traffic Those green, change and clearance intervals in a cycle assigned to any independent movement(s) of traffic.

phase, vehicular A vehicular phase is a phase which is allocated to vehicular traffic movement as timed by the controller unit.

preemption The transfer of the normal control of signals to a special signal control mode for the purpose of servicing railroad crossings, emergency vehicle passage, mass transit vehicle passage, and other special tasks, the control of which require terminating normal traffic control to provide the priority needs of the special task.

preemptor A device or program/routine which provides preemption.

progression The act of various controller units providing specific green indications in accordance with a time schedule to permit continuous operation of groups of vehicles along the street at a planned speed.

red clearance interval A clearance interval which may follow the yellow change interval during which both the terminating phase and the next phase display Red signal indications.

red revert Provision within the controller unit to assure a minimum Red signal indication in a phase following the Yellow Change interval of that phase.

rest The interval portion of a phase when present timing requirements have been completed.

ring A ring consists of two or more sequentially timed and individually selected conflicting phases so arranged as to occur in an established order.

sequence, interval The order of appearance of signal indications during successive intervals of a cycle.

single entry Single entry is a mode of operation (in a multi-ring CU) in which a phase in one ring can be selected and timed alone if there is no demand for service in a nonconflicting phase on the parallel ring(s).

single-ring controller unit A single-ring CU contains two or more sequentially timed and individually selected conflicting phases so arranged as to occur in an established order.

special function A control that may activate specific functions or outputs in an actuated controller unit.

split The segment of the cycle length allocated to each phase or interval that may occur (expressed in seconds). In an actuated controller unit, split is the time in the cycle allocated to a phase.

standby mode An operational state called by master or central command which directs the controller unit to select Pattern, Automatic Flash, or Automatic Free based on local Time Base schedule or Interconnect inputs.

time base control A means for the automatic selection of modes of operation of traffic signals in a manner prescribed by a predetermined time schedule.

timing plan The Split times for all segments (Phase/Interval) of the coordination cycle.

volume The number of vehicles passing a given point per unit of time.

yellow change interval The first interval following the green interval in which the signal indication for that phase is yellow.

yield A command which permits termination of the green interval.

1.4 ABBREVIATIONS AND ACRONYMS

The abbreviations used in this Standard Publication are defined as follows:

BIU	Bus Interface Unit
CA	Controller Assembly
CU	Controller Unit
MMU	Malfunction Management Unit
TBC	Time Base Control
TF	Terminals and Facilities

Section 2
OBJECT DEFINITIONS

This section defines those objects which are specifically used by actuated traffic signal controllers. The objects are defined using the OBJECT-TYPE macro specified in RFC 1212. The text provided from Clause 2.1 through the end of the section (except the clause headings) constitutes the NTCIP Standard ASC MIB.

The clauses below present the objects in lexigraphical order of their OBJECT IDENTIFIERS which correspond to their physical location within the global naming tree. All of the objects defined in this document reside under the "asc" node of the global naming tree. To aid in object management, the "asc" node has been subdivided into logical categories, each defined by a node under the "asc" node. The individual objects are then located under the appropriate node.

Nodes should not be confused with Conformance Groups, which are defined in Annex A. A Conformance Group is a logical grouping of objects which is used for conformance statements. While Conformance Groups will frequently correspond to the nodal structure, a Conformance Group may contain objects which are not lexigraphically ordered. For example, a Schedule Conformance Group contains both "global" and "asc" specific objects.

An object Status of Optional should not be confused with a conformance status of optional or mandatory as defined in Annex A. The object Status of Optional in the MIB means that the object and object definition is current. The status of optional or mandatory in Annex A dictates whether the object is required or not.

Text preceded by a double hyphen in the MIB definitions represent normative text for this standard.

All management applications shall reference the specific device MIB as provided by the device manufacturer for support and constraints (sub-ranges).

2.1 MIB HEADER

```
NTCIP1202-200x DEFINITIONS ::= BEGIN

-- the following OBJECT IDENTIFIERS are used in the ASC MIB:
IMPORTS
    Counter
        FROM RFC1155-SMI
    OBJECT-TYPE
        FROM RFC-1212
    OwnerString, devices
        FROM TMIB-II;


asc OBJECT IDENTIFIER ::= { devices 1 }
```

2.2 PHASE PARAMETERS`

```
phase OBJECT IDENTIFIER
::= { asc 1 }
```

```
-- This node shall contain objects that configure, monitor or
-- control phase functions for this device.
```

2.2.1 Maximum Phases

```
maxPhases   OBJECT-TYPE
   SYNTAX   INTEGER (2..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Phases this
       Actuated Controller Unit supports. This object
       indicates the maximum rows which shall appear in
       the phaseTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxPhases
       <DataConceptType> Data Element
       <Unit> phase"
::= { phase 1 }
```

2.2.2 Phase Table

```
phaseTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF PhaseEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit phase parameters. The number of rows in this
       table is equal to the maxPhases object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.phaseTable
       <DataConceptType> Entity Type"
::= { phase 2 }

phaseEntry   OBJECT-TYPE
   SYNTAX   PhaseEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Parameters for a specific Actuated
       Controller Unit phase.
       <DescriptiveName> NTCIP-1202::ASC.phaseEntry
       <DataConceptType> Entity Type
       <Unit> "
   INDEX   { phaseNumber }
::= { phaseTable 1 }
```

```
PhaseEntry ::= SEQUENCE {
   phaseNumber                INTEGER,
   phaseWalk                  INTEGER,
   phasePedestrianClear       INTEGER,
   phaseMinimumGreen          INTEGER,
   phasePassage               INTEGER,
   phaseMaximum1              INTEGER,
   phaseMaximum2              INTEGER,
   phaseYellowChange          INTEGER,
   phaseRedClear              INTEGER,
   phaseRedRevert             INTEGER,
   phaseAddedInitial          INTEGER,
   phaseMaximumInitial        INTEGER,
   phaseTimeBeforeReduction   INTEGER,
   phaseCarsBeforeReduction   INTEGER,
   phaseTimeToReduce          INTEGER,
   phaseReduceBy              INTEGER,
   phaseMinimumGap            INTEGER,
   phaseDynamicMaxLimit       INTEGER,
   phaseDynamicMaxStep        INTEGER,
   phaseStartup               INTEGER,
   phaseOptions               INTEGER,
   phaseRing                  INTEGER,
   phaseConcurrency           OCTET STRING }
```

2.2.2.1 Phase Number

```
phaseNumber   OBJECT-TYPE
   SYNTAX    INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The phase number for objects in this
       row. This value shall not exceed the maxPhases
       object value.
       <DescriptiveName> NTCIP-1202::ASC.phaseNumber
       <DataConceptType> Data Element
       <Unit> phase"
::= { phaseEntry 1 }
```

2.2.2.2 Phase Walk Parameter

```
phaseWalk   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Walk Parameter in seconds. This
       shall control the amountof time the Walk indication
       shall be displayed.
       <DescriptiveName> NTCIP-1202::ASC.phaseWalk
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.2.a"
::= { phaseEntry 2 }
```

2.2.2.3 Phase Pedestrian Clear Parameter

```
phasePedestrianClear   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Pedestrian Clear Parameter in
       seconds. This shall control the duration of the
       Pedestrian Clearance output (if present) and the
       flashing period of the Don’t Walk output.
       <DescriptiveName> NTCIP-1202::ASC.phasePedestrianClear
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.2.b"
::= { phaseEntry 3 }
```

2.2.2.4 Phase Minimum Green Parameter

```
phaseMinimumGreen   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Minimum Green Parameter in
       seconds (NEMA TS 2 range: 1-255 sec). The first
       timed portion of the Green interval which may be
       set in consideration of the storage of vehicles
       between the zone of detection for the approach
       vehicle detector(s) and the stop line.
       <DescriptiveName> NTCIP-1202::ASC.phaseMinimumGreen
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.1.a.(1)"
::= { phaseEntry 4 }
```

2.2.2.5 Phase Passage Parameter

```
phasePassage   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Passage Parameter in tenth
       seconds (0-25.5 sec). Passage Time, Vehicle Interval,
       Preset Gap, Vehicle Extension: the extensible portion
       of the Green shall be a function of vehicle
       actuations that occur during the Green interval.
       The phase shall remain in the extensible portion of
       the Green interval as long as the passage timer is
       not timed out.  The timing of this portion of the
       green interval shall be reset with each subsequent
       vehicle actuation and shall not commence to time
       again until the vehicle actuation is removed.
       <DescriptiveName> NTCIP-1202::ASC.phasePassage
       <DataConceptType> Data Element
       <Unit> tenth second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.1.a.(2)"
::= { phaseEntry 5 }
```

2.2.2.6 Phase Maximum Green 1 Parameter

```
phaseMaximum1   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Maximum 1 Parameter in seconds
       (NEMA TS 2 range: 1-255 sec). This time setting
       shall determine the maximum length of time this
       phase may be held Green in the presence of a
       serviceable conflicting call. In the absence of a
       serviceable conflicting call the Maximum Green timer
       shall be held reset unless Max Vehicle Recall is
       enabled for this phase.  This is the default maximum
       value to use.  It may be overridden via an external
       input, coordMaximumMode or other method.
       <DescriptiveName> NTCIP-1202::ASC.phaseMaximum1
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1, 3.5.3.2.1.a.(3) and 3.5.3.5"
::= { phaseEntry 6 }
```

2.2.2.7 Phase Maximum Green 2 Parameter

```
phaseMaximum2   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Maximum 2 Parameter in seconds
       (NEMA TS 2 range: 1-255 sec). This time setting
       shall determine the maximum length of time this phase
       may be held Green in the presence of a serviceable
       conflicting call. In the absence of a serviceable
       conflicting call the Maximum Green timer shall be
       held reset unless Max Vehicle Recall is enabled for
       this phase.  This may be implemented as the max green
       timer via an external input, coordMaximumMode or
       other method.
       <DescriptiveName> NTCIP-1202::ASC.phaseMaximum2
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1, 3.5.3.2.1.a.(3), 3.5.3.5
       and 3.5.4.1 (7)"
::= { phaseEntry 7 }
```

2.2.2.8 Phase Yellow Change Parameter

```
phaseYellowChange   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Yellow Change Parameter in tenth
       seconds (NEMA TS 2 range: 3-25.5 sec). Following the
       Green interval of each phase the CU shall provide a
       Yellow Change interval which is timed according to
       the Yellow Change parameter for that phase.
       <DescriptiveName> NTCIP-1202::ASC.phaseYellowChange
       <DataConceptType> Data Element
       <Unit> tenth second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.5.a"
::= { phaseEntry 8 }
```

2.2.2.9 Phase Red Clear Parameter

```
phaseRedClear   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Red Clearance Parameter in tenth
       seconds (0-25.5 sec).Following the Yellow Change
       interval for each phase, the CU shall provide a Red
       Clearance interval which is timed according to the
       Red Clearance parameter for that phase.
       <DescriptiveName> NTCIP-1202::ASC.phaseRedClear
       <DataConceptType> Data Element
       <Unit> tenth second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.5.b"
::= { phaseEntry 9 }
```

2.2.2.10 Phase Red Revert

```
phaseRedRevert OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Red revert time parameter in tenth
       seconds .  A minimum Red indication to be timed
       following the Yellow Change interval and prior to
       the next display of Green on the same signal output
       driver group.

       The unitRedRevert parameter shall act as a minimum
       red revert time for all signal displays.  The
       phaseRedRevert parameter may increase the red revert
       time for a specific phase.  If the phaseRedRevert
       parameter is less than the unitRedRevert the
       unitRedRevert time shall be used.
       <DescriptiveName> NTCIP-1202::ASC.phaseRedRevert
       <DataConceptType> Data Element
       <Unit> tenth second"
::= { phaseEntry 10 }
```

2.2.2.11 Phase Added Initial Parameter

```
phaseAddedInitial   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Added Initial Parameter in tenths
       of seconds (0-25.5 sec). Added Initial parameter
       (Seconds / Actuation) shall determine the time by
       which the variable initial time period will be
       increased from zero with each vehicle actuation
       received during the associated phase Yellow and Red
       intervals.
       <DescriptiveName> NTCIP-1202::ASC.phaseAddedInitial
       <DataConceptType> Data Element
       <Unit> tenth second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.1.b.(1).(b)"
::= { phaseEntry 11 }
```

2.2.2.12 Phase Maximum Initial Parameter

```
phaseMaximumInitial   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Maximum Initial Parameter in
       seconds (0-255 sec). The maximum value of the
       variable initial timing period. Variable Initial
       timing shall equal the lesser of [added initial
       (seconds / actuation) * number of actuations] or
       [ Max Initial ]. The variable initial time shall not
       be less than Minimum Green.
       <DescriptiveName> NTCIP-1202::ASC.phaseMaximumInitial
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.2.1.b.(1).(c)"
::= { phaseEntry 12 }
```

2.2.2.13 Phase Time Before Reduction Parameter

```
phaseTimeBeforeReduction   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Time Before Reduction (TBR)
       Parameter in seconds (0-255 sec). The Time Before
```

```
        Reduction period shall begin when the phase is Green
        and there is a serviceable conflicting call. If the
        serviceable conflicting call is removed before
        completion of this time (or time to reduce), the
        timer shall reset.  Upon completion of the TBR
        period or the CarsBeforeReduction (CBR) parameter is
        satisfied, whichever occurs first, the linear
        reduction of the allowable gap from the Passage Time
        shall begin.
        <DescriptiveName> NTCIP-1202::ASC.phaseTimeBeforeReduction
        <DataConceptType> Data Element
        <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.1.b.(2)"
::= { phaseEntry 13 }
```

2.2.2.14 Phase Cars Before Reduction Parameter

```
phaseCarsBeforeReduction   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Cars Before Reduction (CBR)
       Parameter (0-255 vehicles). When the phase is Green
       and the sum of the cars waiting (vehicle actuations
       during Yellow & Red intervals) on serviceable
       conflicting phases equals or exceeds the CBR
       parameter or the Time Before Reduction (TBR)
       parameter is satisfied, whichever occurs first, the
       linear reduction of the allowable gap from the
       Passage Time shall begin.
       <DescriptiveName> NTCIP-1202::ASC.phaseCarsBeforeReduction
       <DataConceptType> Data Element
       <Unit> vehicle"
::= { phaseEntry 14 }
```

2.2.2.15 Phase Time To Reduce Parameter

```
phaseTimeToReduce   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Time To Reduce Parameter in
       seconds (0-255 sec). This parameter shall control
       the rate of reduction of the allowable gap between
       the Passage Time and Minimum Gap setting.
       <DescriptiveName> NTCIP-1202::ASC.phaseTimeToReduce
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.1.b.(2)"
::= { phaseEntry 15 }
```

2.2.2.16 Phase Reduce By

```
phaseReduceBy   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object may be used for volume
       density gap reduction as an alternate to the linear
       reduction defined by NEMA TS 1 and TS 2.  It contains
       the tenths of seconds to reduce the gap by (0.0 -
       25.5 seconds).  The frequency of reduction shall
       produce the Minimum Gap after a time equal to the
       ‘phaseTimeToReduce’ object.
       <DescriptiveName> NTCIP-1202::ASC.phaseReduceBy
       <DataConceptType> Data Element
       <Unit> tenth second"
::= { phaseEntry 16 }
```

2.2.2.17 Phase Minimum Gap Parameter

```
phaseMinimumGap   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Minimum Gap Parameter in tenth
       seconds (0-25.5 sec). The reduction of the allowable
       gap shall continue until the gap reaches a value
       equal to or less than the minimum gap as set on the
       Minimum Gap control after which the allowable gap
       shall remain fixed at the values set on the Minimum
       Gap control.
       <DescriptiveName> NTCIP-1202::ASC.phaseMinimumGap
       <DataConceptType> Data Element
       <Unit> tenth second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.1 and 3.5.3.2.1.b.(2)"
::= { phaseEntry 17 }
```

2.2.2.18 Phase Dynamic Max Limit

```
phaseDynamicMaxLimit   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall determine either the
       upper or lower limit of the running max in seconds
       (0-255 sec) during dynamic max operation.

       The normal maximum (i.e. Max1, Max2, etc.) shall
       determine the other limit as follows:

       When dynamicMaxLimit is larger than the normal
       maximum, it shall become the upper limit.

       When dynamicMaxLimit is smaller than the normal
       maximum, it shall become the lower limit.

       Setting dynamicMaxLimit greater than zero enables
       dynamic max operation with the normal maximum used
       as the initial maximum setting.  See dynamicMaxStep
       for details on dynamic max operation.

       Maximum recall or a failed detector that is assigned
       to the associated phase shall disable dynamic max
       operation for the phase.
       <DescriptiveName> NTCIP-1202::ASC.phaseDynamicMaxLimit
       <DataConceptType> Data Element
       <Unit> second"
::= { phaseEntry 18 }
```

2.2.2.19 Phase Dynamic Max Step

```
phaseDynamicMaxStep   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall determine the
       automatic adjustment to the running max in tenth
       seconds (0-25.5)

       When a phase maxes out twice in a row, and on each
       successive max out thereafter, one dynamic max step
       value shall be added to the running max until such
       addition would mean the running max was greater than
       the larger of normal max or dynamic max limit.

       When a phase gaps out twice in a row, and on each
       successive gap out thereafter, one dynamic max step
       value shall be subtracted from the running max until
       such subtraction would mean the running max was less
       than the smaller of the normal max or the dynamic max
       limit.

       If a phase gaps out in one cycle and maxes out in
       the next cycle, or vice versa, the running max will
       not change.
       <DescriptiveName> NTCIP-1202::ASC.phaseDynamicMaxStep
       <DataConceptType> Data Element
       <Unit> tenth second"
::= { phaseEntry 19 }
```

2.2.2.20 Phase Startup

```
phaseStartup   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                      phaseNotOn (2),
                      greenWalk (3),
                      greenNoWalk (4),
                      yellowChange (5),
                      redClear (6) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The Phase Startup parameter is an
       enumerated integer which selects the startup state
       for each phase after restoration of a defined power
       interruption or activation of the external start
       input. The following entries are defined:
          other: this phase is not enabled (phaseOptions
             bit 0=0 or phaseRing=0) or initializes in a
             state not defined by this standard.
          phaseNotOn: this phase initializes in a Red state
             (the phase is not active and no intervals are
             timing).
          greenWalk: this phase initializes at the
             beginning of the minimum green and walk timing
             intervals.
          greenNoWalk: this phase initializes at the
             beginning of the minimum green timing interval.
          yellowChange: this phase initializes at the
             beginning of the Yellow Change interval.
          redClear: this phase initializes at the beginning
             of the Red Clearance interval.
       <DescriptiveName> NTCIP-1202::ASC.phaseStartup
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.5.1 and 3.5.5.12"
::= { phaseEntry 20 }
```

2.2.2.21 Phase Options

```
phaseOptions   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Optional phase functions ( 0 = False/
       Disabled, 1 = True/Enabled)
       Bit 15: AddedInitialCalculation - If set (1) the CU
          shall compare counts from all associated
          AddedInitial detectors and use the largest
          count value for the calculations.  If clear (0)
          the CU shall sum all associated AddedInitial
          detector counts and use this sum for the
          calculations.  The ability to modify the setting
          of this bit is optional.
```

```
Bit 14: Conditional Service Enable - in multi-ring
   configurations when set to 1 causes a gapped/maxed
   phase to conditionally service a preceding actuated
   vehicle phase when sufficient time remains before
   max time out of the phase(s) not prepared to
   terminate. Support is optional.
   REFERENCE  NEMA TS 2 Clause 3.5.3.9
Bit 13: Actuated Rest In Walk - when set to 1 causes
   an actuated phase to rest in Walk when there is no
   serviceable conflicting call at the end of Walk
   Timing.
Bit 12: Guaranteed Passage - when set to 1 enables
   an actuated phase operating in volume density mode
   (using gap reduction) to retain the right of
   way for the unexpired portion of the Passage
   time following the decision to terminate the
   green due to a reduced gap.  Support is optional
Bit 11: Simultaneous Gap Disable - in multi-ring
   configurations when set to 1 disables a gapped
   out phase from reverting to the extensible
   portion. Support is optional
   REFERENCE NEMA TS 2 Clause 3.5.5.3
Bit 10: Dual Entry Phase - in multi-ring
   configurations when set to 1 causes the phase to
   become active upon entry into a concurrency group
  (crossing a barrier) when no calls exist in its
   ring within its concurrency group.
   REFERENCE NEMA TS 2 Clause 3.5.5.3
Bit 9: Soft Vehicle Recall - when set to 1 causes a
   call on a phase when all conflicting phases are
   in green dwell or red dwell and there are no
   serviceable conflicting calls.  Support is
   optional.
Bit 8: Ped. Recall - when set to 1 causes a recurring
   pedestrian demand which shall function in the same
   manner as an external pedestrian call except that
   it shall not recycle the pedestrian service until a
   conflicting phase is serviced
   REFERENCE NEMA TS 2 Clause 3.5.3.7
Bit 7: Max Vehicle Recall - when set to 1 causes  a
   call on a phase such that the timing of the Green
   interval for that phase shall be extended to
   Maximum Green time.
   REFERENCE NEMA TS 2 Clause 3.5.3.5
Bit 6: Min. Vehicle Recall - when set to 1 causes
   recurring demand for vehicle service on the phase
   when that phase is not in its Green interval.
   REFERENCE NEMA TS 2 Clause 3.5.3.6
Bit 5: Non Lock Detector Memory - when set to 0 will
   cause the call to be locked at the beginning of
   the yellow interval.  When set to 1 call locking
   will depend on the detectorOptions object.
   REFERENCE NEMA TS 2 Clause 3.5.3.4
Bit 4: Non-Actuated 2 - when set to 1 causes a phase
   to respond to the Call To Non-Actuated 2 input
   (if present) or other method. Support is optional
   REFERENCE NEMA TS 2 Clause 3.5.5.5.8
Bit 3: Non-Actuated 1 - when set to 1 causes a phase
   to respond to the Call To Non-Actuated 1 input
   (if present) or other method. Support is optional
```

```
          REFERENCE NEMA TS 2 Clause 3.5.5.5.8
       Bit 2: Automatic Flash Exit Phase - The CU shall
          move immediately to the beginning of the phase(s)
          programmed as Exit Phase(s) when Automatic Flash
          terminates. Support is optional
          REFERENCE NEMA TS 2 Clause 3.9.1.2.1
       Bit 1: Automatic Flash Entry Phase - When Automatic
          Flash is called, the CU shall service the Entry
          Phase(s), clear to an All Red, then initiate
          flashing operation. Support is optional.
          REFERENCE NEMA TS 2 Clause 3.9.1.2.1
       Bit 0: Enabled Phase - provide a means to define
          whether this phase is used in the current
          configuration. A disabled phase shall not provide
          any outputs nor respond to any phase inputs. The
          object phaseRing = 0 has the same effect.
       <DescriptiveName> NTCIP-1202::ASC.phaseOptions
       <DataConceptType> Data Element"
::= { phaseEntry 21 }
```

2.2.2.22 Phase Ring Parameter

```
phaseRing   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase ring number (1..maxRings) that
       identified the ring which contains the associated
       phase. This value must not exceed the maxRings
       object value. If the ring number is zero, the
       phase is disabled (phaseOptions Bit 0 = 0 has the
       same effect).
       <DescriptiveName> NTCIP-1202::ASC.phaseRing
       <DataConceptType> Data Element
       <Unit> ring"
::= { phaseEntry 22 }
```

2.2.2.23 Phase Concurrency

```
phaseConcurrency   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet contains a phase number
       (binary value) that may run concurrently with the
       associated phase.  Phases that are contained in the
       same ring may NOT run concurrently.
       <DescriptiveName> NTCIP-1202::ASC.phaseConcurrency
       <DataConceptType> Data Element"
::= { phaseEntry 23 }
```

2.2.3 Maximum Phase Groups

```
maxPhaseGroups   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Phase Groups (8
       Phases per group) this Actuated Controller Unit
       supports. This value is equal to TRUNCATE
       [(maxPhases + 7) / 8]. This object indicates the
       maximum rows which shall appear in the
       phaseStatusGroupTable and phaseControlGroupTable.
       <DescriptiveName> NTCIP-1202::ASC.maxPhaseGroups
       <DataConceptType> Data Element
       <Unit> group"
::= { phase 3 }
```

2.2.4 Phase Status Group Table

```
phaseStatusGroupTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF PhaseStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit Phase Output (Red, Yellow, & Green) and Call
       (vehicle & pedestrian) status in groups of eight
       Phases. The number of rows in this table is equal to
       the maxPhaseGroups object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupTable
       <DataConceptType> Entity Type"
::= { phase 4 }

phaseStatusGroupEntry   OBJECT-TYPE
   SYNTAX   PhaseStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Red, Yellow, & Green Output Status and
       Vehicle and Pedestrian Call for eight Actuated
       Controller Unit Phases.
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupEntry
       <DataConceptType> Entity Type"
   INDEX   { phaseStatusGroupNumber }
::= { phaseStatusGroupTable 1 }
```

```
PhaseStatusGroupEntry ::= SEQUENCE {
   phaseStatusGroupNumber       INTEGER,
   phaseStatusGroupReds         INTEGER,
   phaseStatusGroupYellows      INTEGER,
   phaseStatusGroupGreens       INTEGER,
   phaseStatusGroupDontWalks    INTEGER,
   phaseStatusGroupPedClears    INTEGER,
   phaseStatusGroupWalks        INTEGER,
   phaseStatusGroupVehCalls     INTEGER,
   phaseStatusGroupPedCalls     INTEGER,
   phaseStatusGroupPhaseOns     INTEGER,
   phaseStatusGroupPhaseNexts   INTEGER }
```

2.2.4.1 Phase Status Group Number

```
phaseStatusGroupNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Phase StatusGroup number for objects
       in this row. This value shall not exceed the
       maxPhaseGroups object value.
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupNumber
       <DataConceptType> Data Element
       <Unit> group"
::= { phaseStatusGroupEntry 1 }
```

2.2.4.2 Phase Status Group Reds

```
phaseStatusGroupReds   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Red Output Status Mask, when a
       bit = 1, the Phase Red is currently active. When a
       bit = 0, the Phase Red is NOT currently active.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupReds
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 2 }
```

2.2.4.3 Phase Status Group Yellows

```
phaseStatusGroupYellows   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Yellow Output Status Mask, when a
       bit = 1, the Phase Yellow is currently active. When a
       bit = 0, the Phase Yellow is NOT currently active.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupYellows
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 3 }
```

2.2.4.4 Phase Status Group Greens

```
phaseStatusGroupGreens   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Green Output Status Mask, when a
       bit = 1, the Phase Green is currently active. When a
       bit = 0, the Phase Green is NOT currently active.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupGreens
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 4 }
```

2.2.4.5 Phase Status Group Dont Walks

```
phaseStatusGroupDontWalks   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Dont Walk Output Status Mask, when
       a bit = 1, the Phase Dont Walk is currently active.
       When a bit = 0, the Phase Dont Walk is NOT currently
       active.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupDontWalks
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 5 }
```

2.2.4.6 Phase Status Group Pedestrian Clears

```
phaseStatusGroupPedClears   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Ped Clear Output Status Mask, when
        a bit = 1, the Phase Ped Clear is currently active.
        When a bit = 0, the Phase Ped Clear is NOT currently
        active.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupPedClears
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 6 }
```

2.2.4.7 Phase Status Group Walks

```
phaseStatusGroupWalks   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Walk Output Status Mask, when a
       bit = 1, the Phase Walk is currently active. When a
       bit = 0, the Phase Walk is NOT currently active.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupWalks
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 7 }
```

2.2.4.8 Phase Status Group Vehicle Calls

```
phaseStatusGroupVehCalls   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Vehicle Call Status Mask, when a
       bit = 1, the Phase vehicle currently has a call for
       service. When a bit = 0, the Phase vehicle currently
       does NOT have a call for service.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupVehCalls
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 8 }
```

2.2.4.9 Phase Status Group Pedestrian Calls

```
phaseStatusGroupPedCalls   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Pedestrian Call Status Mask, when
       a bit = 1, the  Phase pedestrian currently has a call
       for service. When a bit = 0, the Phase pedestrian
       currently does NOT have a call for service.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupPedCalls
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 9 }
```

2.2.4.10 Phase Status Group Phase Ons

```
phaseStatusGroupPhaseOns   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase On Status Mask, when a bit = 1,
       the Phase is currently active. When a bit = 0, the
       Phase currently is NOT active. The phase is ON
       during the Green, Yellow, & Red Clearance intervals
       of that phase. It shall be permissible for this
       status to be True (bit=1) during the Red Dwell state.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupPhaseOns
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 10 }
```

2.2.4.11 Phase Status Group Phase Nexts

```
phaseStatusGroupPhaseNexts   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Next Status Mask, when a bit = 1,
       the Phase currently is committed to be NEXT in
       sequence & remains present until the phase becomes
       active (On/Timing). When a bit = 0, the Phase
       currently is NOT committed to be NEXT in sequence.
       The phase next to be serviced shall be determined at
       the end of the green interval of the terminating
       phase; except that if the decision cannot be made at
       the end of the Green interval, it shall not be made
       until after the end of all Vehicle Change & Clearance
       intervals.
          Bit 7: Phase # = (phaseStatusGroupNumber * 8)
          Bit 6: Phase # = (phaseStatusGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseStatusGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseStatusGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseStatusGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseStatusGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseStatusGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.phaseStatusGroupPhaseNexts
       <DataConceptType> Data Element"
::= { phaseStatusGroupEntry 11 }
```

2.2.5 Phase Control Table

```
phaseControlGroupTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF PhaseControlGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit Phase Control in groups of eight phases. The
       number of rows in this table is equal to the
       maxPhaseGroups object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupTable
       <DataConceptType> Entity Type
       <Unit> group"
::= { phase 5 }

phaseControlGroupEntry   OBJECT-TYPE
   SYNTAX   PhaseControlGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Control for eight Actuated
       Controller Unit phases.
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupEntry
       <DataConceptType> Entity Type"
   INDEX   { phaseControlGroupNumber }
::= { phaseControlGroupTable 1 }

PhaseControlGroupEntry ::= SEQUENCE {
   phaseControlGroupNumber      INTEGER,
   phaseControlGroupPhaseOmit   INTEGER,
   phaseControlGroupPedOmit     INTEGER,
   phaseControlGroupHold        INTEGER,
   phaseControlGroupForceOff    INTEGER,
   phaseControlGroupVehCall     INTEGER,
   phaseControlGroupPedCall     INTEGER }
```

2.2.5.1 Phase Control Group Number

```
phaseControlGroupNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Phase Control Group number for
       objects in this row. This value shall not exceed the
       maxPhaseGroups object value.
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupNumber
       <DataConceptType> Data Element
       <Unit> group"
::= { phaseControlGroupEntry 1 }
```

2.2.5.2 Phase Omit Control

```
phaseControlGroupPhaseOmit   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to omit phases from being serviced in the
       device. When a bit = 1, the device shall activate
       the System Phase Omit control for that phase. When a
       bit = 0, the device shall not activate the System
       Phase Omit control for that phase.
          Bit 7: Phase # = (phaseControlGroupNumber * 8)
          Bit 6: Phase # = (phaseControlGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseControlGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseControlGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseControlGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseControlGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseControlGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in BACKUP
       Mode. A write to this object shall reset the Backup timer
       to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupPhaseOmit
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.11.2"
::= { phaseControlGroupEntry 2 }
```

2.2.5.3 Pedestrian Omit Control

```
phaseControlGroupPedOmit   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to omit peds from being serviced in the
       device. When a bit = 1, the device shall activate
       the System Ped Omit control for that phase. When a
       bit = 0, the device shall not activate the System
       Ped Omit control for that phase.
          Bit 7: Phase # = (phaseControlGroupNumber * 8)
          Bit 6: Phase # = (phaseControlGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseControlGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseControlGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseControlGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseControlGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseControlGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in BACKUP
       Mode. A write to this object shall reset the Backup timer
       to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupPedOmit
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.11.3"
::= { phaseControlGroupEntry 3 }
```

2.2.5.4 Phase Hold Control

```
phaseControlGroupHold   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to hold phases in the device. When a bit = 1,
       the device shall activate the System Phase Hold
       control for that phase. When a bit = 0, the device
       shall not activate the System Phase Hold control for
       that phase.
          Bit 7: Phase # = (phaseControlGroupNumber * 8)
          Bit 6: Phase # = (phaseControlGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseControlGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseControlGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseControlGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseControlGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseControlGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in BACKUP
       Mode. A write to this object shall reset the Backup timer
       to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupHold
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.11.1"
::= { phaseControlGroupEntry 4 }
```

2.2.5.5 Phase Force Off Control

```
phaseControlGroupForceOff   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to apply force offs
       on a per phase basis. When a bit = 1, the device
       shall activate the System Phase Force Off control
       for that phase. When a bit = 0, the device shall not
       activate the System Phase Force Off control for that
       phase.  When the phase green terminates, the
       associated bit shall be reset to 0.
          Bit 7: Phase # = (phaseControlGroupNumber * 8)
          Bit 6: Phase # = (phaseControlGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseControlGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseControlGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseControlGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseControlGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseControlGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in BACKUP
       Mode. A write to this object shall reset the Backup timer
       to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupForceOff
       <DataConceptType> Data Element"
::= { phaseControlGroupEntry 5 }
```

2.2.5.6 Vehicle Call Control

```
phaseControlGroupVehCall   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to place calls for vehicle service in the
       device. When a bit = 1, the device shall place a
       call for vehicle service on that phase. When a
       bit = 0, the device shall not place a call for
       vehicle service on that phase.
          Bit 7: Phase # = (phaseControlGroupNumber * 8)
          Bit 6: Phase # = (phaseControlGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseControlGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseControlGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseControlGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseControlGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseControlGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in BACKUP
       Mode. A write to this object shall reset the Backup timer
       to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupVehCall
       <DataConceptType> Data Element"
::= { phaseControlGroupEntry 6 }
```

2.2.5.7 Pedestrian Call Control

```
phaseControlGroupPedCall   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to place calls for ped service in the device.
       When a bit = 1, the device shall place a call for ped
       service on that phase. When a bit = 0, the device
       shall not place a call for ped service on that phase.
          Bit 7: Phase # = (phaseControlGroupNumber * 8)
          Bit 6: Phase # = (phaseControlGroupNumber * 8) - 1
          Bit 5: Phase # = (phaseControlGroupNumber * 8) - 2
          Bit 4: Phase # = (phaseControlGroupNumber * 8) - 3
          Bit 3: Phase # = (phaseControlGroupNumber * 8) - 4
          Bit 2: Phase # = (phaseControlGroupNumber * 8) - 5
          Bit 1: Phase # = (phaseControlGroupNumber * 8) - 6
          Bit 0: Phase # = (phaseControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in BACKUP
       Mode. A write to this object shall reset the Backup timer
       to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.phaseControlGroupPedCall
       <DataConceptType> Data Element"
::= { phaseControlGroupEntry 7 }
```

2.3 DETECTOR PARAMETERS

```
detector OBJECT IDENTIFIER
::= { asc 2 }

-- This defines a node for supporting detector objects.
```

2.3.1 Maximum Vehicle Detectors

```
maxVehicleDetectors   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Vehicle Detectors
       this Actuated Controller Unit supports. This object
       indicates the maximum rows which shall appear in the
       vehicleDetectorTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxVehicleDetectors
       <DataConceptType> Data Element
       <Unit> detector"
::= { detector 1 }
```

2.3.2 Vehicle Detector Parameter Table

```
vehicleDetectorTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF VehicleDetectorEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit vehicle detector parameters. The number of rows
       in this table is equal to the maxVehicleDetectors
       object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorTable
       <DataConceptType> Entity Type"
::= { detector 2 }

vehicleDetectorEntry   OBJECT-TYPE
   SYNTAX   VehicleDetectorEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Parameters for a specific Actuated
       Controller Unit detector.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorEntry
       <DataConceptType> Entity Type"
   INDEX   { vehicleDetectorNumber }
::= { vehicleDetectorTable 1 }

VehicleDetectorEntry ::= SEQUENCE {
   vehicleDetectorNumber           INTEGER,
   vehicleDetectorOptions          INTEGER,
   vehicleDetectorCallPhase        INTEGER,
   vehicleDetectorSwitchPhase      INTEGER,
   vehicleDetectorDelay            INTEGER,
   vehicleDetectorExtend           INTEGER,
   vehicleDetectorQueueLimit       INTEGER,
   vehicleDetectorNoActivity       INTEGER,
   vehicleDetectorMaxPresence      INTEGER,
   vehicleDetectorErraticCounts    INTEGER,
   vehicleDetectorFailTime         INTEGER,
   vehicleDetectorAlarms           INTEGER,
   vehicleDetectorReportedAlarms   INTEGER,
   vehicleDetectorReset            INTEGER }
```

2.3.2.1 Vehicle Detector Number

```
vehicleDetectorNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The vehicle detector number for objects
       in this row. The value shall not exceed the
       maxVehicleDetectors object value.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorNumber
       <DataConceptType> Data Element
       <Unit> detector"
::= { vehicleDetectorEntry 1 }
```

2.3.2.2 Vehicle Detector Options Parameter

```
vehicleDetectorOptions   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Vehicle Detector Options Parameter as
       follows (0=Disabled, 1=Enabled):
          Bit 7: Call - if Enabled, the CU shall place
             a demand for vehicular service on the assigned
             phase when the phase is not timing the green
             interval and an actuation is present.
          Bit 6: Queue - if Enabled, the CU shall extend
             the green interval of the assigned phase until a
             gap occurs (no actuation) or until the green
             has been active longer than the
             vehicleDetectorQueueLimit time.
             This is optional.
          Bit 5: AddedInitial - if Enabled, the CU shall
             accumulate detector actuation counts for use in
             the added initial calculations. Counts shall be
             accumulated from the beginning of the yellow
             interval to the beginning of the green interval.
          Bit 4: Passage - if Enabled, the CU shall maintain
             a reset to the associated phase passage timer
             for the duration of the detector actuation when
             the phase is green.
          Bit 3: Red Lock Call - if Enabled, the detector
             will lock a call to the assigned phase if an
             actuation occurs while the phase is not timing
             Green or Yellow. This mode is optional.
          Bit 2: Yellow Lock Call - if Enabled, the detector
             will lock a call to the assigned phase if an
             actuation occurs while the phase is not timing
             Green.
          Bit 1: Occupancy Detector - if Enabled, the
             detector collects data for the associated
             detector occupancy object(s).  This capability
             may not be supported on all detector inputs to
             a device.
          Bit 0: Volume Detector - if Enabled, the detector
             collects data for the associated detector
             volume object(s). This capability may not be
             supported on all detector inputs to a device.
       A SET of both bits 2 & 3 = 1 shall result in bit 2=1 and
       bit 3=0.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorOptions
       <DataConceptType> Data Element"
::= { vehicleDetectorEntry 2}

-- Note: { vehicleDetectorEntry 3} is not used.
```

2.3.2.3 Vehicle Detector Call Phase Parameter

```
vehicleDetectorCallPhase   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object contains assigned phase
       number for the detector input associated with this
       row.  The associated detector call capability is
       enabled when this object is set to a non-zero value.
       The value shall not exceed the value of maxPhases.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorCallPhase
       <DataConceptType> Data Element
       <Unit> phase"
   REFERENCE
      "NEMA TS 2 Clause 3.5.5.5.4 and 3.5.5.5.5"
::= { vehicleDetectorEntry 4 }
```

2.3.2.4 Vehicle Detector Switch Phase Parameter

```
vehicleDetectorSwitchPhase   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Switch Phase Parameter (i.e.,
       Phase Number). The phase to which a vehicle detector
       actuation shall be switched when the assigned phase
       is Yellow or Red and the Switch Phase is Green.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorSwitchPhase
       <DataConceptType> Data Element
       <Unit> phase"
   REFERENCE
      "NEMA TS 2 Clause 3.5.5.5.4.c"
::= { vehicleDetectorEntry 5 }
```

2.3.2.5 Vehicle Detector Delay Parameter

```
vehicleDetectorDelay   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Delay Parameter in tenth
       seconds (0–255.0 sec). The period a detector
       actuation (input recognition) shall be delayed when
       the phase is not Green.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorDelay
       <DataConceptType> Data Element
       <Unit> tenth second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.5.5.4.a"
::= { vehicleDetectorEntry 6 }
```

2.3.2.6 Vehicle Detector Extend Parameter

```
vehicleDetectorExtend   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Extend Parameter in tenth
       seconds (0–25.5 sec). The period a vehicle detector
       actuation (input duration) shall be extended from the
       point of termination , when the phase is Green.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorExtend
       <DataConceptType> Data Element
       <Unit> tenth second"
   REFERENCE
      "NEMA TS 2 Clause 3.5.5.5.4.b"
::= { vehicleDetectorEntry 7 }
```

2.3.2.7 Vehicle Detector Queue Limit

```
vehicleDetectorQueueLimit   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Queue Limit parameter in
       seconds (0-255 sec). The length of time that an
       actuation from a queue detector may continue into
       the phase green.  This time begins when the phase
       becomes green and when it expires any associated
       detector inputs shall be ignored.  This time may be
       shorter due to other overriding device parameters
       (i.e. Maximum time, Force Off’s, ...).
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorQueueLimit
       <DataConceptType> Data Element
       <Unit> second"
::= { vehicleDetectorEntry 8 }
```

2.3.2.8 Vehicle Detector No Activity Parameter

```
vehicleDetectorNoActivity   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector No Activity diagnostic
       Parameter in minutes (0–255 min.) . If an active
       detector does not exhibit an actuation in the
       specified period, it is considered a fault by the
       diagnostics and the detector is classified as Failed.
       A value of 0 for this object shall disable this
       diagnostic for this detector.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorNoActivity
       <DataConceptType> Data Element
       <Unit> minute"
   REFERENCE
      "NEMA TS 2 Clause 3.9.3.1.4.1"
```

```
::= { vehicleDetectorEntry 9 }
```

2.3.2.9 Vehicle Detector Maximum Presence Parameter

```
vehicleDetectorMaxPresence   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Maximum Presence diagnostic
       Parameter in minutes (0-255 min.). If an active
       detector exhibits continuous detection for too long
       a period, it is considered a fault by the diagnostics
       and the detector is classified as Failed. A value of
       0 for this object shall disable this diagnostic for
       this detector.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorMaxPresence
       <DataConceptType> Data Element
       <Unit> minute"
   REFERENCE
      "NEMA TS 2 Clause 3.9.3.1.4.2"
::= { vehicleDetectorEntry 10 }
```

2.3.2.10 Vehicle Detector Erratic Counts Parameter

```
vehicleDetectorErraticCounts   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Erratic Counts diagnostic
       Parameter in counts/minute (0-255 cpm). If an active
       detector exhibits excessive actuations, it is
       considered a fault by the diagnostics and the
       detector is classified as Failed. A value of 0 for
       this object shall disable this diagnostic for this
       detector.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorErraticCounts
       <DataConceptType> Data Element
       <Unit> count"
   REFERENCE
      "NEMA TS 2 Clause 3.9.3.1.4.3"
::= { vehicleDetectorEntry 11 }
```

2.3.2.11 Vehicle Detector Fail Time Parameter

```
vehicleDetectorFailTime   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Fail Time in seconds (0..255 sec).
       If a detector diagnostic indicates that the
       associated detector input is failed, then a call
       shall be placed on the associated phase during all
       non-green intervals. When each green interval begins
       the call shall be maintained for the length of time
       specified by this object and then removed. If the
       value of this object equals the maximum value (255)
       then a constant call shall be placed on the
       associated phase (max recall). If the value of this
       object equals zero then no call shall be placed on
       the associated phase for any interval (no recall).
       Compliant devices may support a limited capability
       for this object (i.e. only max recall or max recall
       and no recall). At a minimum the max recall setting
       must be supported.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorFailTime
       <DataConceptType> Data Element
       <Unit> second"
::= { vehicleDetectorEntry 12 }
```

2.3.2.12 Vehicle Detector Alarms

```
vehicleDetectorAlarms   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall return indications of
       detector alarms. Detector Alarms are indicated as
       follows:
          Bit 7: Other Fault - The detector has failed due
             to some other cause.
          Bit 6: Reserved.
          Bit 5: Reserved.
          Bit 4: Configuration Fault - Detector is assigned
             but is not supported.
          Bit 3: Communications Fault - Communications to
             the device (if present) have failed.
          Bit 2: Erratic Output Fault - This detector has
             been flagged as non-operational due to erratic
             outputs (excessive counts) by the CU detector
             diagnostic.
          Bit 1: Max Presence Fault - This detector has been
             flagged as non-operational due to a presence
             indicator that exceeded the maximum expected
             time by the CU detector diagnostic.
          Bit 0: No Activity Fault - This detector has been
             flagged as non-operational due to lower than
             expected activity by the CU detector diagnostic.
       Once set a bit shall maintain its state as long as the
       condition exists. The bit shall clear when the condition
       no longer exists.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorAlarms
       <DataConceptType> Data Element"
::= { vehicleDetectorEntry  13 }
```

2.3.2.13 Vehicle Detector Reported Alarms

```
vehicleDetectorReportedAlarms   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall return detector device
       reported alarms (via some communications mechanism).
       Inductive Loop Detector Alarms are indicated as
       follows:
          Bit 7: Reserved.
          Bit 6: Reserved.
          Bit 5: Reserved.
          Bit 4: Excessive Change Fault - This detector has
             been flagged as non-operational due to an
             inductance change that exceeded expected values.
          Bit 3: Shorted Loop Fault - This detector has been
             flagged as non-operational due to a shorted loop
             wire.
          Bit 2: Open Loop Fault - This detector has been
             flagged as non-operational due to an open loop
             (broken wire).
          Bit 1: Watchdog Fault - This detector has been
             flagged as non-operational due to a watchdog
             error.
          Bit 0: Other - This detector has been flagged as
             non-operational due to some other error.
       Once set a bit shall maintain its state as long as
       the condition exists.  The bit shall clear when the
       condition no longer exists.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorReportedAlarms
       <DataConceptType> Data Element"
::= { vehicleDetectorEntry  14 }
```

2.3.2.14 Vehicle Detector Reset

```
vehicleDetectorReset   OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object when set to TRUE (one)
       shall cause the CU to command the associated detector
       to reset.  This object shall automatically return to
       FALSE (zero)  after the CU has issued the reset
       command.

       NOTE: this may affect other detector (detector
       channels) that are physically attached to a common
       reset line.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorReset
       <DataConceptType> Data Element"
::= { vehicleDetectorEntry 15 }
```

2.3.3 Maximum Vehicle Detector Status Groups

```
maxVehicleDetectorStatusGroups   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The maximum number of detector status
       groups (8 detectors per group) this device supports.
       This value is equal to TRUNCATE
       [(maxVehicleDetectors + 7 ) / 8].  This object
       indicates the maximum number of rows which shall
       appear in the vehicleDetectorStatusGroupTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxVehicleDetectorStatusGroups
       <DataConceptType> Data Element
       <Unit> group"
::= { detector 3 }
```

2.3.4 Vehicle Detector Status Group Table

```
vehicleDetectorStatusGroupTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF VehicleDetectorStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing detector status in
       groups of eight detectors. The number of rows in this
       table is equal to the maxVehicleDetectorStatusGroups
       object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorStatusGroupTable
       <DataConceptType> Entity Type"
::= { detector 4 }

vehicleDetectorStatusGroupEntry   OBJECT-TYPE
   SYNTAX   VehicleDetectorStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A group (row) of detector status.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorStatusGroupEntry
       <DataConceptType> Entity Type"
   INDEX  { vehicleDetectorStatusGroupNumber }
::= { vehicleDetectorStatusGroupTable  1 }

VehicleDetectorStatusGroupEntry ::= SEQUENCE {
   vehicleDetectorStatusGroupNumber   INTEGER,
   vehicleDetectorStatusGroupActive   INTEGER,
   vehicleDetectorStatusGroupAlarms   INTEGER }
```

2.3.4.1 Detector Status Group Number

```
vehicleDetectorStatusGroupNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The detector status group number for
       objects in this row. This value shall not exceed the
       maxVehicleDetectorStatusGroups object value.
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorStatusGroupNumber
       <DataConceptType> Data Element
       <Unit> group"
::= { vehicleDetectorStatusGroupEntry 1 }
```

2.3.4.2 Detector Status Group Active

```
vehicleDetectorStatusGroupActive   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall return the detection
       status of each detector associated with the group.
       Each detector shall be represented as ON (detect) or
       OFF (no-detect) by individual bits in this object.
       If a detector is ON then the associated bit shall be
       set (1).  If a detector is OFF then the associated
       bit shall be clear (0).
          Bit 7: Det # = ( vehicleDetectorStatusGroupNumber * 8)
          Bit 6: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 1
          Bit 5: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 2
          Bit 4: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 3
          Bit 3: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 4
          Bit 2: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 5
          Bit 1: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 6
          Bit 0: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorStatusGroupActive
       <DataConceptType> Data Element"
::= { vehicleDetectorStatusGroupEntry 2 }
```

2.3.4.3 Detector Alarm Status

```
vehicleDetectorStatusGroupAlarms   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall return the alarm status
       of the detectors associated with the group.  Each
       detector alarm status shall be represented as ON or
       OFF by individual bits in this object.  If any
       detector alarm (defined in the vehicleDetectorAlarm
       object) is active the associated bit shall be
       set (1).  If a detector alarm is not active the
       associated bit shall be clear (0).
```

```
          Bit 7: Det # = ( vehicleDetectorStatusGroupNumber * 8)
          Bit 6: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 1
          Bit 5: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 2
          Bit 4: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 3
          Bit 3: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 4
          Bit 2: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 5
          Bit 1: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 6
          Bit 0: Det # = ( vehicleDetectorStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.vehicleDetectorStatusGroupAlarms
       <DataConceptType> Data Element"
::= { vehicleDetectorStatusGroupEntry 3 }
```

2.3.5 Volume / Occupancy Report

```
volumeOccupancyReport   OBJECT IDENTIFIER
::= { detector 5 }

-- This node contains the objects necessary to support volume /
-- occupancy reporting.
```

2.3.5.1 Volume / Occupancy Sequence

```
volumeOccupancySequence   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines a Sequence Number
       for Volume/Occupancy data collection.  This object
       is used to detect duplicate or missing reports.  The
       value cycles within the limits of 0 to 255.  This
       object is incremented by one at the expiration of the
       volumeOccupancyPeriod time.
       <DescriptiveName> NTCIP-1202::ASC.volumeOccupancySequence
       <DataConceptType> Data Element
       <Unit> sequence"
::= { volumeOccupancyReport 1 }
```

2.3.5.2 Volume / Occupancy Period

```
volumeOccupancyPeriod   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the number of
       seconds (0-255 sec) that comprise the Volume/Occupancy
       collection period.  When the collection period
       expires the device shall increment the
       volumeOccupancySequence,  update the
       volumeOccupancyTable entries and reset the volume
       occupancy timer.
       <DescriptiveName> NTCIP-1202::ASC.volumeOccupancyPeriod
       <DataConceptType> Data Element
       <Unit> second"
::= { volumeOccupancyReport 2 }
```

2.3.5.3 Active Volume / Occupancy Detectors

```
activeVolumeOccupancyDetectors   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The number of detectors in this device.
       This object indicates how many rows are in the
       volumeOccupancyTable object. There shall be a row
       for every detector that is collecting volume or
       occupancy data (refer to detectorOptions in the
       detectorTable).
       <DescriptiveName> NTCIP-1202::ASC.activeVolumeOccupancyDetectors
       <DataConceptType> Data Element
       <Unit> detector"
::= { volumeOccupancyReport 3 }
```

2.3.5.4 Volume / Occupancy Table

```
volumeOccupancyTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF VolumeOccupancyEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Detector Volume and
       Occupancy data collected. The number of rows in this
       table is equal to the activeVolumeOccupancyDetectors
       object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.volumeOccupancyTable
       <DataConceptType> Entity Type"
::= { volumeOccupancyReport 4 }

volumeOccupancyEntry   OBJECT-TYPE
   SYNTAX   VolumeOccupancyEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> The Volume and Occupancy data collected
       for one of the detectors in the device.
       <DescriptiveName> NTCIP-1202::ASC.volumeOccupancyEntry
       <DataConceptType> Entity Type"
   INDEX   { vehicleDetectorNumber }
::= { volumeOccupancyTable 1 }

VolumeOccupancyEntry ::= SEQUENCE {
   detectorVolume      INTEGER,
   detectorOccupancy   INTEGER }
```

2.3.5.4.1 Volume Data

```
detectorVolume   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Volume data collected over the
       volumeOccupancyPeriod. This value shall range from
       0 to 254 indicating the volume of traffic crossing
       the associated detectorNumber during the collection
       period.

       The value 255 shall indicate volume overflow.
       <DescriptiveName> NTCIP-1202::ASC.detectorVolume
       <DataConceptType> Data Element
       <Unit> volume"
::= { volumeOccupancyEntry 1 }
```

2.3.5.4.2 Occupancy Data

```
detectorOccupancy   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Detector Occupancy as a percentage of the
       volumeOccupancyPeriod over which the data was collected
       or Detector Unit Diagnostic Information.  The value of
       the object shall indicate occupancy or detector diagnostic
       information as follows:
          Range   Meaning
          0-200   Detector Occupancy in 0.5% Increments
        201-209   Reserved
          210     Max Presence Fault
          211     No Activity Fault
          212     Open loop Fault
          213     Shorted loop Fault
          214     Excessive Change Fault
          215     Reserved
          216     Watchdog Fault
          217     Erratic Output Fault
        218-255   Reserved

       Faults shall be indicated for all collection periods
       during which a fault is detected if either occupancy
       data or volume data is being collected.  The highest
       numbered fault shall be presented if more than one
       fault is active (i.e. indicate OpenLoop rather than
       NoActivity).
       <DescriptiveName> NTCIP-1202::ASC.detectorOccupancy
       <DataConceptType> Data Element
       <Unit> occupancy"
::= { volumeOccupancyEntry 2 }
```

2.3.6 Maximum Pedestrian Detectors

```
maxPedestrianDetectors OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Pedestrian
       Detectors this Actuated Controller Unit supports.
       This object indicates the maximum rows which shall
       appear in the pedestrianDetectorTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxVehicleDetectors
       <DataConceptType> Data Element
       <Unit> detector"
::= { detector 6 }
```

2.3.7 Pedestrian Detector Parameter Table

```
pedestrianDetectorTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF PedestrianDetectorEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit pedestrian detector parameters. The number of
       rows in this table is equal to the
       maxPedestrianDetectors object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorTable
       <DataConceptType> Entity Type"
::= { detector 7 }

pedestrianDetectorEntry   OBJECT-TYPE
   SYNTAX   PedestrianDetectorEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Parameters for a specific Actuated
       Controller Unit pedestrian detector.
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorEntry
       <DataConceptType> Entity Type"
   INDEX   { pedestrianDetectorNumber }
::= { pedestrianDetectorTable 1 }

PedestrianDetectorEntry ::= SEQUENCE {
   pedestrianDetectorNumber          INTEGER,
   pedestrianDetectorCallPhase       INTEGER,
   pedestrianDetectorNoActivity      INTEGER,
   pedestrianDetectorMaxPresence     INTEGER,
   pedestrianDetectorErraticCounts   INTEGER,
   pedestrianDetectorAlarms          INTEGER }
```

2.3.7.1 Pedestrian Detector Number

```
pedestrianDetectorNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The pedestrianDetector number for
       objects in this row. The value shall not exceed the
       maxPedestrianDetectors object value.
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorNumber
       <DataConceptType> Data Element
       <Unit> detector"
::= { pedestrianDetectorEntry 1 }
```

2.3.7.2 Pedestrian Detector Call Phase Parameter

```
pedestrianDetectorCallPhase   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object contains assigned phase
       number for the pedestrian detector input associated
       with this row.  The associated detector call
       capability is enabled when this object is set to a
       non-zero value. The value shall not exceed the
       value of maxPhases.
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorCallPhase
       <DataConceptType> Data Element
       <Unit> phase"
::= { pedestrianDetectorEntry 2 }
```

2.3.7.3 Pedestrian Detector No Activity Parameter

```
pedestrianDetectorNoActivity   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Pedestrian Detector No Activity
       diagnostic Parameter in minutes (0–255 min.) . If an
       active detector does not exhibit an actuation in the
       specified period, it is considered a fault by the
       diagnostics and the detector is classified as Failed.
       A value of 0 for this object shall disable this
       diagnostic for this detector.
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorNoActivity
       <DataConceptType> Data Element
       <Unit> minute"
   REFERENCE
      "NEMA TS 2 Clause 3.9.3.1.4.1"
::= { pedestrianDetectorEntry 3 }
```

2.3.7.4 Pedestrian Detector Maximum Presence Parameter

```
pedestrianDetectorMaxPresence   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Pedestrian Detector Maximum Presence
       diagnostic Parameter in minutes (0-255 min.). If an
       active detector exhibits continuous detection for
       too long a period, it is considered a fault by the
       diagnostics and the detector is classified as Failed.
       A value of 0 for this object shall disable this
       diagnostic for this detector.
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorMaxPresence
       <DataConceptType> Data Element
       <Unit> minute"
   REFERENCE
      "NEMA TS 2 Clause 3.9.3.1.4.2"
::= { pedestrianDetectorEntry 4 }
```

2.3.7.5 Pedestrian Detector Erratic Counts Parameter

```
pedestrianDetectorErraticCounts   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Pedestrian Detector Erratic Counts
       diagnostic Parameter in counts/minute (0-255 cpm). If
       an active detector exhibits excessive actuations, it
       is considered a fault by the diagnostics and the
       detector is classified as Failed. A value of 0 for
       this object shall disable this diagnostic for this
       detector.
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorErraticCounts
       <DataConceptType> Data Element
       <Unit> count"
   REFERENCE
      "NEMA TS 2 Clause 3.9.3.1.4.3"
::= { pedestrianDetectorEntry 5 }
```

2.3.7.6 Pedestrian Detector Alarms

```
pedestrianDetectorAlarms  OBJECT-TYPE
   SYNTAX  INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall return indications of
       detector alarms. Detector Alarms are indicated as
       follows:
          Bit 7: Other Fault - The detector has failed due
             to some other cause.
          Bit 6: Reserved.
          Bit 5: Reserved.
          Bit 4: Configuration Fault - Detector is assigned
```

```
             but is not supported.
          Bit 3: Communications Fault - Communications to
             the device (if present) have failed.
          Bit 2: Erratic Output Fault - This detector has
             been flagged as non-operational due to erratic
             outputs (excessive counts) by the CU detector
             diagnostic.
          Bit 1: Max Presence Fault - This detector has been
             flagged as non-operational due to a presence
             indicator that exceeded the maximum expected
             time by the CU detector diagnostic.
          Bit 0: No Activity Fault - This detector has been
             flagged as non-operational due to lower than
             expected activity by the CU detector diagnostic
       Once set a bit shall maintain its state as long as the
       condition exists.  The bit shall clear when the condition
       no longer exists.
       <DescriptiveName> NTCIP-1202::ASC.pedestrianDetectorAlarms
       <DataConceptType> Data Element"
::= { pedestrianDetectorEntry  6 }
```

2.4 UNIT PARAMETERS

```
unit OBJECT IDENTIFIER
::= { asc 3 }

--This defines a node for supporting unit objects.
```

2.4.1 Startup Flash Parameter

```
unitStartUpFlash   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Unit Start up Flash time parameter in
       seconds (0 to 255 sec). The period/state (Start-Up
       Flash occurs when power is restored following a
       device defined power interruption. During the
       Start-Up Flash state, the Fault Monitor and Voltage
       Monitor outputs shall be inactive (if present).
       <DescriptiveName> NTCIP-1202::ASC.unitStartUpFlash
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.9.1.1"
::= { unit 1 }
```

2.4.2 Automatic Ped Clear Parameter

```
unitAutoPedestrianClear   OBJECT-TYPE
   SYNTAX   INTEGER { disable(1),
                      enable (2) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Unit Automatic Ped Clear parameter
       (1 = False/Disable 2=True/Enable). When enabled,
       the CU shall time the Pedestrian Clearance interval
       when Manual Control Enable is active and prevent the
       Pedestrian Clearance interval from being terminated
       by the Interval Advance input.
       <DescriptiveName> NTCIP-1202::ASC.unitAutoPedestrianClear
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.3.10"
::= { unit 2 }
```

2.4.3 Backup Time Parameter

```
unitBackupTime   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The Backup Time in seconds (0-65535 sec).
       When any of the defined system control parameters
       is SET, the backup timer is reset. After reset it
       times the unitBackupTime interval. If the unitBackupTime
       interval expires without a SET operation to any of
       the system control parameters, then the CU shall
       revert to Backup Mode. A value of zero (0) for this
       object shall disable this feature. The system control
       parameters are:
          phaseControlGroupPhaseOmit,
          phaseControlGroupPedOmit,
          phaseControlGroupHold,
          phaseControlGroupForceOff,
          phaseControlGroupVehCall,
          phaseControlGroupPedCall,
          systemPatternControl,
          systemSyncControl,
          preemptControlState,
          ringControlGroupStopTime,
          ringControlGroupForceOff,
          ringControlGroupMax2,
          ringControlGroupMaxInhibit,
          ringControlGroupPedRecycle,
          ringControlGroupRedRest,
          ringControlGroupOmitRedClear,
          unitControl,
          specialFunctionOutputState (deprecated), and
          specialFunctionOutputControl.
       <DescriptiveName> NTCIP-1202::ASC.unitBackupTime
       <DataConceptType> Data Element
       <Unit> second"
::= { unit 3 }
```

2.4.4 Unit Red Revert Parameter

```
unitRedRevert   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The red revert in tenth seconds ( 0.0 -
       25.5 sec). This value shall provide the minimum
       red revert time for all phases (i.e. if it is
       greater than a phaseRedRevert object value, then
       this value shall be used as the red revert time for
       the affected phase). This object provides a minimum
       Red indication following the Yellow Change interval
       and prior to the next display of Green on the same
       signal output driver group.
       <DescriptiveName> NTCIP-1202::ASC.unitRedRevert
       <DataConceptType> Data Element
       <Unit> tenth second"
::= { unit 4 }
```

2.4.5 Unit Control Status

```
unitControlStatus   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                      systemControl (2),
                      systemStandby (3),
                      backupMode(4),
                      manual (5),
                      timebase (6),
                      interconnect (7),
                      interconnectBackup (8)}
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Control Mode for Pattern, Flash, or
       Free at the device:
          other: control by a source other than those
             listed here.
          systemControl: control by master or central
             commands.
          systemStandby: control by local based on master or
             central command to use local control.
          backupMode: Backup Mode (see Terms).
          manual: control by entry other than zero in
             coordOperationalMode.
          timebase: control by the local Time Base.
          interconnect: control by the local Interconnect
             inputs.
          interconnectBackup: control by local TBC due to
             invalid Interconnect inputs or loss of sync.
       <DescriptiveName> NTCIP-1202::ASC.unitControlStatus
       <DataConceptType> Data Element"
::= { unit 5 }
```

2.4.6 Unit Flash Status

```
unitFlashStatus   OBJECT-TYPE
   SYNTAX   INTEGER { other(1),
                      notFlash(2),
                      automatic(3),
                      localManual(4),
                      faultMonitor(5),
                      mmu(6),
                      startup(7),
                      preempt (8)}
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Flash modes:
         other: the CU is in flash for some other reason.
         notFlash: the CU is not in Flash
         automatic: the CU is currently in an Automatic
            Flash state.
         localManual: the Controller Unit Local Flash input
            is active, MMU Flash input is not active, and
            Flash is not commanded by the Master.
         faultMonitor: the CU is currently in a Fault
            Monitor State.
         mmu: the Controller Unit MMU Flash input is active
            and the CU is not in Start-Up Flash.
         startup: the CU is currently timing the Start-Up
            Flash period.
         preempt: the CU is currently timing the preempt
            Flash.
       <DescriptiveName> NTCIP-1202::ASC.unitFlashStatus
       <DataConceptType> Data Element"
::= { unit 6 }
```

2.4.7 Unit Alarm Status 2

```
unitAlarmStatus2   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Device Alarm Mask 2 ( 0 = False,
       1 = True) as follows:
          Bit 7: Reserved.
          Bit 6: Reserved.
          Bit 5: Offset Transitioning - Whenever the CU is
             performing an offset transition (correction in
             process)
          Bit 4: Stop Time - When either CU Stop Time Input
             becomes active.
          Bit 3: External Start - When the CU External Start
             becomes active.
          Bit 2: Response Fault - When any NEMA TS2 Port 1
             response frame fault occurs.
          Bit 1: Low Battery - When any battery voltage
             falls below the required level.
          Bit 0: Power Restart - When power returns after a
             power interruption.
       Once set, a bit shall maintain it’s state as long as
       the condition exists. Bit 0 (Power Restart) status
       shall be maintained until a READ of this object
       occurs.
       <DescriptiveName> NTCIP-1202::ASC.unitAlarmStatus2
       <DataConceptType> Data Element"
::= { unit 7 }
```

2.4.8 Unit Alarm Status 1

```
unitAlarmStatus1   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Device Alarm Mask 1 ( 0 = False,
       1 = True) as follows:
          Bit 7: CoordActive - When coordination is active
             and not preempted or overridden.
          Bit 6: Local Free - When any of the CU inputs and/or
             programming cause it not to run coordination.
          Bit 5: Local Flash - When the Controller Unit
             Local Flash input becomes active, MMU Flash
             input is not active, and Flash is not commanded
             by the system.
          Bit 4: MMU Flash - When the Controller Unit MMU
             Flash input remains active for a period of time
             exceeding the Start-Up Flash time.
          Bit 3: Cycle Fail - When a local Controller Unit
             is operating in the non-coordinated mode,
             whether the result of a Cycle Fault or Free
             being the current normal mode, and cycling
             diagnostics indicate that a serviceable call
             exists that has not been serviced for two
             cycles.
          Bit 2: Coord Fail - When a Coord Fault is in
             effect and a Cycle Fault occurs again within
             two cycles of the coordination retry.
          Bit 1: Coord Fault - When a Cycle Fault is in
             effect and the serviceable call has been
             serviced within two cycles after the Cycle
             Fault.
          Bit 0: Cycle Fault - When the Controller Unit is
             operating in the coordinated mode and cycling
             diagnostics indicate that a serviceable call
             exists that has not been serviced for two
             cycles.
       Once set, a bit shall maintain it’s state as long as
       the condition exists.
       <DescriptiveName> NTCIP-1202::ASC.unitAlarmStatus1
       <DataConceptType> Data Element"
::= { unit 8 }
```

2.4.9 Short Alarm Status

```
shortAlarmStatus   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Short Alarm Mask ( 0 = False,
       1 = True) as follows:
          Bit 7: Critical Alarm - When the Stop Time input
             is active.
          Bit 6: Non-Critical Alarm - When an physical alarm
             input is active.
          Bit 5: Detector Fault - When any detectorAlarm
             fault occurs.
          Bit 4: Coordination Alarm - When the CU is not
             running the called pattern without offset
             correction within three cycles of the command.
             An offset correction requiring less than three
             cycles due to cycle overrun caused by servicing
             a pedestrian call shall not cause a
             Coordination Alarm.
          Bit 3: Local Override - When any of the CU inputs
             and/or programming cause it not to run coordination.
          Bit 2: Local Cycle Zero - When running coordinated
             and the Coord Cycle Status (coordCycleStatus)
             has passed through zero.
          Bit 1: T&F Flash - When either the Local Flash or
             MMU Flash input becomes active.
          Bit 0: Preempt - When any of the CU Preempt
             inputs become active.
       Once set, a bit shall maintain it’s state as long as
       the condition exists. Bit 2 (Local Cycle Zero) status
       shall be maintained until a READ of this object
       occurs.
       <DescriptiveName> NTCIP-1202::ASC.shortAlarmStatus
       <DataConceptType> Data Element"
::= { unit 9 }
```

2.4.10 Unit Control

```
unitControl   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to activate unit functions in the device
       ( 0 = False / Disabled, 1 = True / Enabled) as
       follows:
          Bit 7: Dimming Enable - when set to 1, causes
             channel dimming to operate as configured. For
             dimming to occur, (this control OR a dimming
             input) AND a 'timebaseAscAuxillaryFunction'
             must be True.
             REFERENCE NEMA TS 2 Clause 3.9.2
          Bit 6: Interconnect - when set to 1, shall cause
             the interconnect inputs to operate at a higher
             priority than the timebase control (TBC On
             Line).
             REFERENCE NEMA TS 2 Clause 3.6.2.3 and 3.8.3
          Bit 5: Walk Rest Modifier - when set to 1, causes
             non-actuated phases to remain in the timed-out
             Walk state (rest in Walk) in the absence of a
             serviceable conflicting call.
             REFERENCE NEMA TS 2 Clause 3.5.5.5.13
          Bit 4: Call to Non-Actuated 2 - when set to 1,
             causes any phase(s) appropriately programmed
             in the phaseOptions object to operate in the
             Non-Actuated Mode.
             REFERENCE NEMA TS 2 Clause 3.5.5.5.8
          Bit 3: Call to Non-Actuated 1 - when set to 1,
             causes any phase(s) appropriately programmed
             in the phaseOptions object to operate in the
             Non-Actuated Mode.
             REFERENCE NEMA TS 2 Clause 3.5.5.5.8
          Bit 2: External Minimum Recall - when set to 1,
             causes a recurring demand on all vehicle phases
             for a minimum vehicle service.
             REFERENCE NEMA TS 2 Clause 3.5.5.5.9
          Bit 1: Reserved
          Bit 0: Reserved
       When a bit = 1, the device shall activate the Unit
       control. When a bit = 0, the device shall not
       activate the Unit control.

       A SET of a 'reserved' bit to a value other than
       zero (0) shall return a badValue(3) error.

       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       BACKUP timer (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.unitControl
       <DataConceptType> Data Element"
::= { unit 10 }
```

2.4.11 Maximum Alarm Groups

```
maxAlarmGroups   OBJECT-TYPE
   SYNTAX   INTEGER(1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object contains the maximum number
       of alarm groups (8 alarm inputs per group) this
       device supports.  This object indicates the maximum
       rows which shall appear in the alarmGroupTable
       object.
       <DescriptiveName> NTCIP-1202::ASC.maxAlarmGroups
       <DataConceptType> Data Element
       <Unit> alarm Group"
::= { unit 11 }
```

2.4.12 Alarm Group Table

```
alarmGroupTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF AlarmGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> This table contains alarm input status
       in groups of eight inputs. The number of rows in
       this table is equal to the maxAlarmGroups object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.alarmGroupTable
       <DataConceptType> Entity Type"
::= { unit 12 }

alarmGroupEntry   OBJECT-TYPE
   SYNTAX   AlarmGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Status for eight alarm inputs.
       <DescriptiveName> NTCIP-1202::ASC.alarmGroupEntry
       <DataConceptType> Entity Type"
   INDEX  { alarmGroupNumber }
::= { alarmGroupTable 1 }

AlarmGroupEntry::= SEQUENCE {
   alarmGroupNumber   INTEGER,
   alarmGroupState    INTEGER}
```

2.4.12.1 Alarm Group Number

```
alarmGroupNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The alarm group number for objects in
       this row.  This value shall not exceed the
       maxAlarmGroups object value.
       <DescriptiveName> NTCIP-1202::ASC.alarmGroupNumber
       <DataConceptType> Data Element
       <Unit> group"
::= { alarmGroupEntry 1 }
```

2.4.12.2 Alarm Group State

```
alarmGroupState   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Alarm input state bit field.  When a
       bit = 1, the associated physical alarm input is
       active.  When a bit = 0, the associated alarm input
       is NOT active.
          Bit 7: Alarm Input # = ( alarmGroupNumber * 8)
          Bit 6: Alarm Input # = ( alarmGroupNumber * 8) -1
          Bit 5: Alarm Input # = ( alarmGroupNumber * 8) -2
          Bit 4: Alarm Input # = ( alarmGroupNumber * 8) -3
          Bit 3: Alarm Input # = ( alarmGroupNumber * 8) -4
          Bit 2: Alarm Input # = ( alarmGroupNumber * 8) -5
          Bit 1: Alarm Input # = ( alarmGroupNumber * 8) -6
          Bit 0: Alarm Input # = ( alarmGroupNumber * 8) -7
       <DescriptiveName> NTCIP-1202::ASC.alarmGroupState
       <DataConceptType> Data Element"
::= {alarmGroupEntry 2 }
```

2.4.13 Maximum Special Function Outputs

```
maxSpecialFunctionOutputs   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Special Functions
       this Actuated Controller Unit supports.
       <DescriptiveName> NTCIP-1202::ASC.maxSpecialFunctionOutputs
       <DataConceptType> Data Element
       <Unit> output"
::= { unit 13 }
```

2.4.14 Special Function Output Table

```
specialFunctionOutputTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF SpecialFunctionOutputEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit special function output objects. The number of
       rows in this table is equal to the
       maxSpecialFunctionOutputs object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.specialFunctionOutputTable
       <DataConceptType> Entity Type"
::= { unit 14 }

specialFunctionOutputEntry   OBJECT-TYPE
   SYNTAX   SpecialFunctionOutputEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Control for  Actuated Controller Unit
       system special functions.
       <DescriptiveName> NTCIP-1202::ASC.specialFunctionOutputEntry
       <DataConceptType> Entity Type"
   INDEX   { specialFunctionOutputNumber }
::= { specialFunctionOutputTable 1 }

SpecialFunctionOutputEntry ::= SEQUENCE {
   specialFunctionOutputNumber   INTEGER,
--   specialFunctionOutputState    INTEGER, deprecated
   specialFunctionOutputControl  INTEGER,
   specialFunctionOutputStatus   INTEGER }
```

2.4.14.1 Special Function Output Number

```
specialFunctionOutputNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The special function output number associated with object
       in this row. This value shall not exceed the maxSpecialFunctionOutputs
       object value.
       <DescriptiveName> NTCIP-1202::ASC.specialFunctionOutputNumber
       <DataConceptType> Data Element
       <Unit> output"
::= { specialFunctionOutputEntry 1 }
```

2.4.14.2 Special Function Output State

```
-- { specialFunctionOutputEntry  2 } is deprecated ... see Annex D
```

2.4.14.3 Special Function Output Control

```
specialFunctionOutputControl   OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The special function output (logical or
       physical) in the device may be controlled by this
       object.
             0 = OFF & 1 = ON
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       BACKUP timer (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.specialFunctionOutputControl
       <DataConceptType> Data Element"
::= { specialFunctionOutputEntry  3 }
```

2.4.14.4 Special Function Output Status

```
specialFunctionOutputStatus   OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The current status (ON-OFF) of the
       special function output (logical or physical) in the
       device.
             0 = OFF & 1 = ON
       <DescriptiveName> NTCIP-1202::ASC.specialFunctionOutputStatus
       <DataConceptType> Data Element"
::= { specialFunctionOutputEntry  4 }
```

2.5 COORDINATION PARAMETERS

```
coord   OBJECT IDENTIFIER
::= { asc 4 }

-- The coord node contains objects that support coordination
-- configuration, status and control functions for the device.
```

2.5.1 Coord Operational Mode Parameter

```
coordOperationalMode   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the operational
       mode for coordination. The possible modes are:
         Value   Description
           0     Automatic - this mode provides for coord
                 operation, free, and flash to be determined
                 automatically by the possible sources (i.e.
                 Interconnect, Time Base, or System Commands).
          1-253  Manual Pattern - these modes provides for
                 Coord operation running this pattern.  This
                 selection of pattern overrides all other
                 pattern commands.
          254    Manual Free - this mode provides for Free
                 operation without coordination or Automatic
                 Flash from any source.
          255    Manual Flash - this mode provides for
                 Automatic Flash without coordination or
                 Free from any source.
       <DescriptiveName> NTCIP-1202::ASC.coordOperationalMode
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.6.2.4"
::= { coord 1 }
```

2.5.2 Coord Correction Mode Parameter

```
coordCorrectionMode   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                      dwell (2),
                      shortway (3),
                      addOnly (4) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the Coord Correction
       Mode. The possible modes are:
         other: the coordinator establishes a new offset by
            a mechanism not defined in this standard.
         dwell: when changing offset, the coordinator shall
            establish a new offset by dwelling in the coord
            phase(s) until the desired offset is reached.
         shortway (Smooth): when changing offset, the
            coordinator shall establish a new offset by
            adding or subtracting to/from the timings in a
            manner that limits the cycle change. This
            operation is performed in a device specific
            manner.
         addOnly: when changing offset, the coordinator
            shall establish a new offset by adding to the
            timings in a manner that limits the cycle
            change. This operation is performed in a device
            specific manner.
       <DescriptiveName> NTCIP-1202::ASC.coordCorrectionMode
       <DataConceptType> Data Element"
::= { coord 2 }
```

2.5.3 Coord Maximum Mode Parameter

```
coordMaximumMode   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                      maximum1 (2),
                      maximum2 (3),
                      maxInhibit (4) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "This object defines the Coord Maximum Mode. The possible
       modes are:
          other: the maximum mode is determined by some
             other mechanism not defined in this standard.
          maximum1: the internal Maximum 1 Timing shall be
             effective while coordination is running a
             pattern.
          maximum2: the internal Maximum 2 Timing shall be
             effective while coordination is running a
             pattern.
          maxInhibit: the internal Maximum Timing shall be
             inhibited while coordination is running a
             pattern.
       <DescriptiveName> NTCIP-1202::ASC.coordMaximumMode
       <DataConceptType> Data Element"
::= { coord 3 }
```

2.5.4 Coord Force Mode Parameter

```
coordForceMode   OBJECT-TYPE
   SYNTAX   INTEGER { other(1),
                      floating (2),
                      fixed (3) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the Pattern Force
       Mode. The possible modes are:
          other: the CU implements a mechanism not defined
             in this standard.
          floating: each non-coord phase will be forced to
             limit its time to the split time value. This allows
             unused split time to revert to the coord phase.
          fixed: each non-coord phase will be forced at a fixed
             position in the cycle. This allows unused
             split time to revert to the following phase.
       <DescriptiveName> NTCIP-1202::ASC.coordForceMode
       <DataConceptType> Data Element"
::= { coord 4 }
```

2.5.5 Maximum Patterns Parameter

```
maxPatterns   OBJECT-TYPE
   SYNTAX   INTEGER (1..253)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The maximum number of Patterns this
       Actuated Controller Unit supports. This object
       indicates how many rows are in the patternTable
       object (254 and 255 are defined as non-pattern
       status for Free and Flash).
       <DescriptiveName> NTCIP-1202::ASC.maxPatterns
       <DataConceptType> Data Element
       <Unit> pattern"
::= { coord 5 }
```

2.5.6 Pattern Table Type

```
patternTableType   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                     patterns (2),
                     offset3 (3),
                     offset5 (4) }
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object provides information about
       any special organizational structure required for
       the pattern table. The defined structures are as
       follows:
          other: The pattern table setup is not described
             in this standard, refer to device manual.
          patterns: Each row of the pattern table
             represents a unique pattern and has no
             dependencies on other rows.
          offset3: The pattern table is organized into
             plans which have three offsets.  Each plan
             uses three consecutive rows.  Only
             patternOffsetTime and patternSequenceNumber
             values may vary between each of the three rows.
             Plan 1 is contained in rows 1, 2 and 3, Plan 2
             is contained in rows 4, 5 and 6, Plan 3 is in
             rows 7, 8 and 9, etc.
          offset5:  The pattern table is organized into
             plans which have five offsets.  Each plan
             occupies five consecutive rows.  Only
             patternOffsetTime and patternSequenceNumber
             values may vary between each of the rows.
             Plan 1 is contained in rows 1, 2, 3, 4 and
             5, Plan 2 is contained in rows 6, 7, 8, 9 and
             10, Plan 3 is contained in rows 11, 12, 13, 14
             and 15, etc.
       <DescriptiveName> NTCIP-1202::ASC.patternTableType
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.6.2.1 and 3.6.2.2"
::= { coord 6 }
```

2.5.7 Pattern Table

```
patternTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF PatternEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit coordination Pattern parameters. The number of
       rows in this table is equal to the maxPatterns object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.patternTable
       <DataConceptType> Entity Type"
::= { coord 7 }

patternEntry   OBJECT-TYPE
   SYNTAX   PatternEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Parameters for a specific Actuated
       Controller Unit pattern.
       <DescriptiveName> NTCIP-1202::ASC.patternEntry
       <DataConceptType> Entity Type"
   INDEX   { patternNumber }
::= { patternTable 1 }

PatternEntry ::= SEQUENCE {
   patternNumber           INTEGER,
   patternCycleTime        INTEGER,
   patternOffsetTime       INTEGER,
   patternSplitNumber      INTEGER,
   patternSequenceNumber   INTEGER }
```

2.5.7.1 Pattern Number Entry

```
patternNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..253)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The pattern number for objects in this
       row. This value shall not exceed the maxPatterns
       object value.
       <DescriptiveName> NTCIP-1202::ASC.patternNumber
       <DataConceptType> Data Element
       <Unit> pattern"
::= { patternEntry 1 }
```

2.5.7.2 Pattern Cycle Time

```
patternCycleTime   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The patternCycleTime object specifies
       the length of the pattern cycle in seconds (NEMA
       TS 2 range: 30-255).  A pattern cycle time less than
       adequate to service the minimum requirements of all
       phases shall result in Free mode. While this condition
       exists, the Local Free bit of unitAlarmStatus1 and the
       Local Override bit of shortAlarmStatus shall be set to
       one (1).

       The minimum requirements of a phase with a not-actuated
       ped include Minimum Green, Walk, Pedestrian Clear, Yellow
       Clearance, and Red Clearance; the minimum requirements of
       a phase with an actuated pedestrian include Minimum Green,
       Yellow Clearance, and Red Clearance. If the pattern cycle
       time is zero and the associated split table (if any)
       contains values greater than zero, then the CU shall
       utilize the split time values as maximum values for each
       phase.
       <DescriptiveName> NTCIP-1202::ASC.patternCycleTime
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.6.2.1.1"
::= { patternEntry 2 }
```

2.5.7.3 Pattern Offset Time Parameter

```
patternOffsetTime   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The patternOffsetTime defines by how
       many seconds (NEMA TS 2 range: 0-254) the local time
       zero shall lag the system time zero (synchronization
       pulse) for this pattern. An offset value equal to
       or greater than the patternCycleTime shall result in Free
       being the operational mode.  While this condition
       exists, the Local Free bit of unitAlarmStatus1 and
       the LocalOverride bit of shortAlarmStatus shall be
       set to one (1).
       <DescriptiveName> NTCIP-1202::ASC.patternOffsetTime
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.6.2.2"
::= { patternEntry 3 }
```

2.5.7.4 Pattern Split Number Parameter

```
patternSplitNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to locate
       information in the splitTable to use for this
       pattern. This value shall not exceed the maxSplits
       object value.
       <DescriptiveName> NTCIP-1202::ASC.patternSplitNumber
       <DataConceptType> Data Element
       <Unit> split"
::= { patternEntry 4 }
```

2.5.7.5 Pattern Sequence Number Parameter

```
patternSequenceNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to locate
       information in the sequenceTable to use with this
       pattern. This value shall not exceed the
       maxSequences object value.
       <DescriptiveName> NTCIP-1202::ASC.patternSequenceNumber
       <DataConceptType> Data Element
       <Unit> sequence"
::= { patternEntry 5 }
```

2.5.8 Maximum Splits

```
maxSplits   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The maximum number of Split Plans this
       Actuated Controller Unit supports. This object
       indicates how many Split plans are in the splitTable
       object.
       <DescriptiveName> NTCIP-1202::ASC.maxSplits
       <DataConceptType> Data Element
       <Unit> split"
::= { coord 8 }
```

2.5.9 Split Table

```
splitTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF SplitEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit coordination split parameters. The number of
       rows in this table is equal to maxSplits.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.splitTable
       <DataConceptType> Entity Type"
::= { coord 9 }

splitEntry   OBJECT-TYPE
   SYNTAX   SplitEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Split type Parameters for a specific
       Actuated Controller Unit phase.
       <DescriptiveName> NTCIP-1202::ASC.splitEntry
       <DataConceptType> Entity Type"
   INDEX   { splitNumber, splitPhase }
::= { splitTable 1 }

SplitEntry ::= SEQUENCE {
   splitNumber       INTEGER,
   splitPhase        INTEGER,
   splitTime         INTEGER,
   splitMode         INTEGER,
   splitCoordPhase   INTEGER }
```

2.5.9.1 Split Number

```
splitNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The object defines which rows of the
       split table comprise a split group. All rows that
       have the same splitNumber are in the same split
       group. The value of this object shall not exceed the
       maxSplits object value.
       <DescriptiveName> NTCIP-1202::ASC.splitNumber
       <DataConceptType> Data Element
       <Unit> split"
::= { splitEntry 1 }
```

2.5.9.2 Split Phase Number

```
splitPhase   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The phase number for objects in this
       row. The value of this object shall not exceed the
       maxPhases object value.
       <DescriptiveName> NTCIP-1202::ASC.splitPhase
       <DataConceptType> Data Element
       <Unit> phase"
::= { splitEntry 2 }
```

2.5.9.3 Split Time Parameter

```
splitTime   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The time in seconds the splitPhase is
       allowed to receive (i.e. before a Force Off is
       applied) when constant demands exist on all phases.
       In floating coordForceMode, this is always the
       maximum time a non-coordinated phase is allowed to
       receive.  In fixed coordForceMode, the actual allowed
       time may be longer if a previous phase gapped out.

       The splitTime includes all phase clearance times for
       the associated phase. The split time shall be longer
       than the sum of the phase minimum service
       requirements for the phase. When the time is NOT
       adequate to service the minimum service requirements
       of the phase, Free Mode shall be the result. The
       minimum requirements of a phase with a not-actuated
       ped include Minimum Green, Walk, Pedestrian Clear,
       Yellow Clearance, and Red Clearance; the minimum
       requirements of a phase with an actuated pedestrian
       include Minimum Green, Yellow Clearance, and Red
       Clearance.

       If the cycleTime entry of the associated
       patternTable entry is zero (i.e. the device is in
       Free Mode), then the value of this object shall be
       applied, if non-zero, as a maximum time for the
       associated phase.

       If the critical path through the phase diagram is
       less than the cycleTime entry of the associated
       patternTable entry, all extra time is alloted to the
       coordination phase in each ring.

       If the critical path through the phase diagram is
       greater than the cycleTime entry of the associated
       patternTable entry (and the cycleTime is not zero)
       the device shall operate in the Free Mode.

       While the Free Mode condition exists, the Local
       Override bit of shortAlarm shall be set to one (1).
       <DescriptiveName> NTCIP-1202::ASC.splitTime
       <DataConceptType> Data Element
       <Unit> second"
   REFERENCE
      "NEMA TS 2 Clause 3.6.2.1.2"
::= { splitEntry 3 }
```

2.5.9.4 Split Mode Parameter

```
splitMode   OBJECT-TYPE
   SYNTAX   INTEGER { other(1),
                      none (2),
                      minimumVehicleRecall (3),
                      maximumVehicleRecall (4),
                      pedestrianRecall (5),
                      maximumVehicleAndPedestrianRecall (6),
                      phaseOmitted (7) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines operational
       characteristics of the phase. The following options
       are available:
         other: the operation is not specified in this
            standard
         none: no split mode control.
         minimumVehicleRecall: this phase operates with a
            minimum vehicle recall.
         maximumVehicleRecall: this phase operates with a
            maximum vehicle recall.
         pedestrianRecall: this phase operates with a
            pedestrian recall.
         maximumVehicleAndPedestrianRecall: this phase
            operates with a maximum vehicle & pedestrian
            recall.
         phaseOmitted: this phase is omitted.
       <DescriptiveName> NTCIP-1202::ASC.splitMode
       <DataConceptType> Data Element"
::= { splitEntry 4 }
```

2.5.9.5 Split Coordinated Phase

```
splitCoordPhase  OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> To select the associated phase as a
       coordinated phase this object shall be set to TRUE
       (non zero).
       <DescriptiveName> NTCIP-1202::ASC.splitCoordPhase
       <DataConceptType> Data Element"
::= { splitEntry 5 }
```

2.5.10 Coordination Pattern Status

```
coordPatternStatus   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the running
       coordination pattern/mode in the device. The
       possible values are:
         Value Description
           0   Not used
         1-253 Pattern - indicates the currently running
               pattern
          254  Free - indicates Free operation without
               coordination.
          255  Flash - indicates Automatic Flash without
               coordination.
       <DescriptiveName> NTCIP-1202::ASC.coordPatternStatus
       <DataConceptType> Data Element"
::= { coord 10 }
```

2.5.11 Local Free Status

```
localFreeStatus   OBJECT-TYPE
SYNTAX   INTEGER { other(1),
                   notFree(2),
                   commandFree(3),
                   transitionFree(4),
                   inputFree(5),
                   coordFree(6),
                   badPlan(7),
                   badCycleTime(8),
                   splitOverrun (9),
                   invalidOffset (10),
                   failed(11) }
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Free modes:
         other: Some other condition has caused the device
            to run in free mode.
         notFree: The unit is not running in free mode.
         commandFree: the current pattern command is the
            Free mode pattern.
         transitionFree: the CU has a pattern command but
            is cycling to a point to begin coordination.
         inputFree: one of the CU inputs cause it to not
            respond to coordination.
         coordFree: the CU programming for the called
            pattern is to run Free.
         badPlan: Free - the called pattern is invalid.
         badCycleTime: the pattern cycle time is less than
            adequate to service the minimum requirements of
            all phases.
         splitOverrun: Free - the sum of the critical path
            splitTime’s exceed the programmed
            patternCycleTime value.
         invalidOffset: Free - reserved / not used
         failed: cycling diagnostics have called for Free.

       An ASC may provide diagnostics beyond those stated
       herein. Therefore, for a set of given bad data, the
       free status between devices may be inconsistent.
       <DescriptiveName> NTCIP-1202::ASC.localFreeStatus
       <DataConceptType> Data Element"
::= { coord 11 }
```

2.5.12 Coordination Cycle Status

```
coordCycleStatus   OBJECT-TYPE
   SYNTAX   INTEGER (0..510)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Coord Cycle Status represents the
       current position in the local coord cycle of the
       running pattern (0 to 510 sec). This value normally
       counts down from patternCycleTime to Zero. This
       value may exceed the patternCycleTime during a coord
       cycle with offset correction (patternCycleTime +
       correction).
       <DescriptiveName> NTCIP-1202::ASC.coordCycleStatus
       <DataConceptType> Data Element
       <Unit> second"
::= { coord 12 }
```

2.5.13 Coordination Sync Status

```
coordSyncStatus   OBJECT-TYPE
   SYNTAX   INTEGER (0..510)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Coord Sync Status represents the
       time since the system reference point for the
       running pattern (0 to 510 sec). This value normally
       counts up from Zero to the next system reference
       point (patternCycleTime). This value may exceed the
       patternCycleTime during a coord cycle in which the
       system reference point has changed.
       <DescriptiveName> NTCIP-1202::ASC.coordSyncStatus
       <DataConceptType> Data Element
       <Unit> second"
::= { coord 13 }
```

2.5.14 System Pattern Control

```
systemPatternControl   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to establish the
       Called System Pattern/Mode for the device. The
       possible values are:
         Value   Description
           0     Standby - the system relinquishes control
                 of the device.
         1-253   Pattern - these values indicate the system
                 commanded pattern
          254    Free - this value indicates a call for Free
          255    Flash - this value indicates a call for
                 Automatic Flash
       If an unsupported / invalid pattern is called, Free
       shall be the operational mode.
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.systemPatternControl
       <DataConceptType> Data Element"
::= { coord 14 }
```

2.5.15 System Sync Control

```
systemSyncControl   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to establish the
       system reference point for the Called System Pattern
       by providing the current position in the system
       pattern cycle (0-254 sec). The device shall
       recognize a write to this object as a command to
       establish the time until the next system reference
       point. Thereafter, the system reference point shall
       be assumed to occur at a frequency equal to the
       patternCycleTime.

       When the value in the object is 255, the system
       reference point shall be referenced to the local
       Time Base in accordance with its programming.

       This CU must maintain an accuracy of 0.1 seconds
       based on the receipt of the SET packet.
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.systemSyncControl
       <DataConceptType> Data Element
       <Unit> second"
::= { coord  15 }
```

2.6 TIME BASE PARAMETERS

```
timebaseAsc OBJECT IDENTIFIER
::= { asc 5 }

-- This object is an identifier used to group all objects for
-- support of timebase functions.  If a device implements timebase
-- functions then these objects shall be supported.
```

2.6.1 Time Base Pattern Sync Parameter

```
timebaseAscPatternSync   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Pattern Sync Reference in minutes past
       midnight.  When the value is 65535, the controller
       unit shall use the Action time as the Sync Reference
       for that pattern. Action time is the hour and minute
       associated with the active dayPlanEventNumber (as
       defined in NTCIP 1201).
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscPatternSync
       <DataConceptType> Data Element
       <Unit> minute"
   REFERENCE
      "NEMA TS 2 Clause 3.8.2"
::= { timebaseAsc 1 }
```

2.6.2 Maximum Time Base Actions

```
maxTimebaseAscActions   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Actions this
       device supports. This object indicates the maximum
       rows which shall appear in the
       timebaseAscActionTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxTimebaseAscActions
       <DataConceptType> Data Element
       <Unit> action"
::= { timebaseAsc 2 }
```

2.6.3 Time Base Asc Action Table

```
timebaseAscActionTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF TimebaseAscActionEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit Time Base action parameters. The number of rows
       in this table is equal to the maxTimebaseAscActions
       object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscActionTable
       <DataConceptType> Entity Type"
::= { timebaseAsc 3 }

timebaseAscActionEntry   OBJECT-TYPE
   SYNTAX   TimebaseAscActionEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Action Parameters for a Actuated
       Controller Unit Time Base Program.
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscActionEntry
       <DataConceptType> Entity Type"
   INDEX   { timebaseAscActionNumber }
::= { timebaseAscActionTable 1 }

TimebaseAscActionEntry ::= SEQUENCE {
   timebaseAscActionNumber        INTEGER,
   timebaseAscPattern             INTEGER,
   timebaseAscAuxillaryFunction   INTEGER,
   timebaseAscSpecialFunction     INTEGER }
```

2.6.3.1 Time Base Action Number

```
timebaseAscActionNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The time base Action number for objects
       in this row. This value shall not exceed the
       maxTimebaseAscActions object value.
         This object may be defined as a dayPlanActionOID (as
       defined in NTCIP 1201).
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscActionNumber
       <DataConceptType> Data Element
       <Unit> action"
::= { timebaseAscActionEntry 1 }
```

2.6.3.2 Time Base Action Pattern Parameter

```
timebaseAscPattern   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The Pattern that shall be active when
       this Action is active. The value shall not exceed
       the value of maxPatterns, except for flash or free.  A pattern
       of zero indicates that no pattern is being selected.
       A pattern = 0 relinquishes control to entity of a
       lower priority than timebase and allows that entity
       to control (i.e., interconnect if available).
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscPattern
       <DataConceptType> Data Element
       <Unit> pattern"
::= { timebaseAscActionEntry 2 }
```

2.6.3.3 Time Base Action Auxiliary Function Parameter

```
timebaseAscAuxillaryFunction   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The Auxiliary functions that shall be
       active when this Action is active.
          Bit 7: Reserved
          Bit 6: Reserved
          Bit 5: Reserved
          Bit 4: Reserved
          Bit 3: Dimming enabled if set (non-zero),
             disabled if clear (zero). For dimming to
             occur, this control AND ('unitControl' OR a
             dimming input) must be True.
          Bit 2: Auxiliary Function 3 enabled if set
             (non-zero), disabled if clear (zero).
          Bit 1: Auxiliary Function 2 enabled if set
             (non-zero), disabled if clear (zero).
          Bit 0: Auxiliary Function 1 enabled if set
             (non-zero), disabled if clear (zero).
       A SET of a 'reserved' bit to a value other than
       zero (0) shall return a badValue(3) error.
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscAuxillaryFunction
       <DataConceptType> Data Element"
::= { timebaseAscActionEntry 3 }
```

2.6.3.4 Time Base Action Special Function Parameter

```
timebaseAscSpecialFunction   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The Special Functions that shall be
       active when this Action is active.
          Bit 7: Special Function 8
          Bit 6: Special Function 7
          Bit 5: Special Function 6
          Bit 4: Special Function 5
          Bit 3: Special Function 4
          Bit 2: Special Function 3
          Bit 1: Special Function 2
          Bit 0: Special Function 1
       Bit = 0 - False/Disabled, Bit = 1 - True/Enabled
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscSpecialFunction
       <DataConceptType> Data Element"
::= { timebaseAscActionEntry 4 }
```

2.6.4 Time Base Asc Action Status

```
timebaseAscActionStatus   OBJECT-TYPE
   SYNTAX   INTEGER(0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object indicates the current time
       base Action Table row that will be used when the CU
       is in Time Base operation. A value of zero indicates
       that no time base Action is selected.
       <DescriptiveName> NTCIP-1202::ASC.timebaseAscActionStatus
       <DataConceptType> Data Element"
::= { timebaseAsc 4 }
```

2.7 PREEMPT PARAMETERS

```
preempt OBJECT IDENTIFIER
::= { asc 6 }

-- The preempt node contains objects that support preempt input
-- functions for the device.
```

2.7.1 Maximum Preempts

```
maxPreempts   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Preempts this
       Actuated Controller Unit supports. This object
       indicates the maximum rows which shall appear in the
       preemptTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxPreempts
       <DataConceptType> Data Element
       <Unit> preempt"
   REFERENCE
      "NEMA TS 2 Clause 3.7"
::= { preempt 1 }
```

2.7.2 Preempt Table

```
preemptTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF PreemptEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit preemption parameters. The number of rows in
       this table is equal to the maxPreempts object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.preemptTable
       <DataConceptType> Entity Type"
::={ preempt 2 }

preemptEntry   OBJECT-TYPE
   SYNTAX   PreemptEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Parameters for a specific Actuated
       Controller Unit preemptor.
       <DescriptiveName> NTCIP-1202::ASC.preemptEntry
       <DataConceptType> Entity Type"
   INDEX    { preemptNumber }
::={ preemptTable 1}

PreemptEntry ::= SEQUENCE {
   preemptNumber            INTEGER,
   preemptControl           INTEGER,
   preemptLink              INTEGER,
   preemptDelay             INTEGER,
   preemptMinimumDuration   INTEGER,
   preemptMinimumGreen      INTEGER,
   preemptMinimumWalk       INTEGER,
   preemptEnterPedClear     INTEGER,
   preemptTrackGreen        INTEGER,
   preemptDwellGreen        INTEGER,
   preemptMaximumPresence   INTEGER,
   preemptTrackPhase        OCTET STRING,
   preemptDwellPhase        OCTET STRING,
   preemptDwellPed          OCTET STRING,
   preemptExitPhase         OCTET STRING,
   preemptState             INTEGER,
   preemptTrackOverlap      OCTET STRING,
   preemptDwellOverlap      OCTET STRING,
   preemptCyclingPhase      OCTET STRING,
   preemptCyclingPed        OCTET STRING,
   preemptCyclingOverlap    OCTET STRING,
   preemptEnterYellowChange INTEGER,
   preemptEnterRedClear     INTEGER,
   preemptTrackYellowChange INTEGER,
   preemptTrackRedClear     INTEGER }
```

2.7.2.1 Preempt Number

```
preemptNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The preempt number for objects in this
       row. The value shall not exceed the maxPreempts
       object value.
         When all preemptControl objects have a value where
       bit 2 = 0, each preemptNumber routine shall be a higher
       priority and override all preemptNumber routines that
       have a larger preemptNumber.
         When a preemptControl object has a value where
       bit 2 = 1, the next higher preemptNumber becomes of
       equal priority with the preemptNumber but may still be
       a higher priority than larger preemptNumbers depending
       on bit 2 of the relavent preemptControl objects.
       <DescriptiveName> NTCIP-1202::ASC.preemptNumber
       <DataConceptType> Data Element
       <Unit> preempt"
::= { preemptEntry 1 }
```

2.7.2.2 Preempt Control Parameter

```
preemptControl   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Preempt Miscellaneous Control Parameter
       Mask ( Bit=0: False/Disabled, Bit=1: True/Enabled)
       as follows:
          Bit 7: Reserved
          Bit 6: Reserved
          Bit 5: Reserved
          Bit 4: Reserved
          Bit 3: Flash Dwell - the CU shall cause the
             phases listed in the preemptDwellPhase object
             to flash Yellow during the Dwell interval.  All
             active phases not listed in preemptDwellPhase
             shall flash Red.
               The CU shall cause the overlaps listed in
             the preemptDwellOverlap object to flash Yellow
             during the Dwell state. All active overlaps not
             listed in preemptDwellOverlap shall flash Red.
             Preempt cycling phase programming is ignored if
             this bit is set.
             This control is optional.
          Bit 2: Preempt Override preemptNumber + 1 -
             provide a means to define whether this preempt
             shall NOT override the next higher numbered
             Preempt.  When set (1) this preempt shall not
             override the next higher numbered preempt.
             Lowered numbered preempts override higher
             numbered preempts. For example, 1 overrides 3,
             and the only way to get 3 equal to 1, is to set
             both 1 and 2 to NOT override the next higher
             numbered preempt.  This parameter shall be
             ignored when preemptNumber equals maxPreempts.
          Bit 1: Preempt Override Flash - provide a means
             to define whether this preempt shall NOT
             override Automatic Flash. When set (1) this
             preempt shall not override Automatic Flash.
          Bit 0: Non-Locking Memory - provide a means to
             enable an operation which does not require
             detector memory. When set (1) a preempt
             sequence shall not occur if the preempt input
             terminates prior to expiration of the
             preemptDelay time.

       A SET of a 'reserved' bit to a value other than
       zero (0) shall return a badValue(3) error.
       <DescriptiveName> NTCIP-1202::ASC.preemptControl
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.7.2.1 and 3.7.2.2"
   DEFVAL   { 0 }
::= { preemptEntry 2 }
```

2.7.2.3 Preempt Link Parameter

```
preemptLink   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object provides a means to define
       a higher priority preempt to be combined (linked)
       with this preempt. At the end of preemptDwellGreen, the
       linked preempt shall receive an automatic call that
       shall be maintained as long as the demand for this
       preempt is active. Any value that is not a higher
       priority preempt or a valid preempt shall be ignored.
       The value shall not exceed the maxPreempts object
       value.
       <DescriptiveName> NTCIP-1202::ASC.preemptLink
       <DataConceptType> Data Element
       <Unit> preempt"
   DEFVAL   { 0 }
::= { preemptEntry 3 }
```

2.7.2.4 Preempt Delay Parameter

```
preemptDelay   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Preempt Delay Time in seconds (0-600
       sec). This value determines the time the preempt
       input shall be active prior to initiating any preempt
       sequence.  A non-locking preempt input which is
       removed prior to the completion of this time shall
       not cause a preempt sequence to occur.
       <DescriptiveName> NTCIP-1202::ASC.preemptDelay
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 0 }
::= { preemptEntry 4 }
```

2.7.2.5 Preempt Duration Parameter

```
preemptMinimumDuration   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Preempt Minimum Duration Time in seconds
       (0..65535 sec). This value determines the minimum time
       during which the preempt is active. Duration begins
       timing at the end of Preempt Delay (if non zero) and
       will prevent an exit from the Dwell interval until
       this time has elapsed.
       <DescriptiveName> NTCIP-1202::ASC.preemptMinimumDuration
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 0 }
::= { preemptEntry 5 }
```

2.7.2.6 Preempt Minimum Green Parameter

```
preemptMinimumGreen   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Preempt Minimum Green Time in seconds
       (0-255 sec). A preempt initiated transition shall
       not cause the termination of an existing Green prior
       to its display for lesser of the phase’s Minimum
       Green time or this period. CAUTION - if this value
       is zero, phase Green is terminated immediately.
       <DescriptiveName> NTCIP-1202::ASC.preemptMinimumGreen
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 255 }
::= { preemptEntry 6 }
```

2.7.2.7 Preempt Minimum Walk Parameter

```
preemptMinimumWalk   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Preempt Minimum Walk Time in seconds
       (0-255 sec). A preempt initiated transition shall
       not cause the termination of an existing Walk prior
       to its display for the lesser of the phase’s Walk
       time or this period. CAUTION - if this value is
       zero, phase Walk is terminated immediately.
       <DescriptiveName> NTCIP-1202::ASC.preemptMinimumWalk
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 255 }
::= { preemptEntry 7 }
```

2.7.2.8 Preempt Enter Pedestrian Clear Parameter

```
preemptEnterPedClear   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Enter Ped ClearTime in seconds (0-255
       sec). This parameter controls the ped clear timing
       for a normal Walk signal terminated by a preempt
       initiated transition.  A preempt initiated
       transition shall not cause the termination of a
       Pedestrian Clearance prior to its display for the
       lesser of the phase’s Pedestrian Clearance time or
       this period. CAUTION - if this value is zero, phase
       Ped Clear is terminated immediately.
       <DescriptiveName> NTCIP-1202::ASC.preemptEnterPedClear
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 255 }
::= { preemptEntry 8 }
```

2.7.2.9 Preempt Track Green Parameter

```
preemptTrackGreen   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Track Clear Green Time in seconds
```

```
       (0-255 sec). This parameter controls the green
       timing for the track clearance movement. Track
       Clear phase(s) are enabled in the preemptTrackPhase
       object. If this value is zero, the track clearance
       movement is omitted, regardless of preemptTrackPhase
       programming.
       <DescriptiveName> NTCIP-1202::ASC.preemptTrackGreen
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 0 }
::= { preemptEntry 9 }
```

2.7.2.10 Preempt Minimum Dwell Parameter

```
preemptDwellGreen   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Minimum Dwell interval in seconds (1-255
       sec). This parameter controls the minimum timing for
       the dwell interval. Phase(s) active during the
       Dwell interval are enabled in preemptDwellPhase
       and preemptCyclingPhase objects.

       The Dwell interval shall not terminate prior to the
       completion of preemptMinimumDuration, preemptDwellGreen
       (this object), and the call is no longer present.
       <DescriptiveName> NTCIP-1202::ASC.preemptDwellGreen
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 10 }
::= { preemptEntry 10 }
```

2.7.2.11 Preempt Maximum Presence Parameter

```
preemptMaximumPresence   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Preempt Maximum Presence time in
       seconds (0-65535 sec). This value determines the
       maximum time which a preempt call may remain active
       and be considered valid.  When the preempt call has
       been active for this time period, the CU shall
       return to normal operation.  This preempt call shall
       be considered invalid until such time as a change in
       state occurs (no longer active).  When set to zero
       the preempt maximum presence time is disabled.
       <DescriptiveName> NTCIP-1202::ASC.preemptMaximumPresence
       <DataConceptType> Data Element
       <Unit> second"
   DEFVAL   { 0 }
::= { preemptEntry 11 }
```

2.7.2.12 Preempt Track Phase Parameter

```
preemptTrackPhase   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a phaseNumber(binary value) that shall be
       active during the Preempt Track Clear intervals.
       The values of phaseNumber used here shall not
       exceed maxPhases or violate the Consistency Checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptTrackPhase
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 12 }
```

2.7.2.13 Preempt Dwell Phase Parameter

```
preemptDwellPhase   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a phaseNumber (binary value) that
       specifies the phase(s) to be served in the
       Preempt Dwell interval. The phase(s) defined in
       preemptCyclingPhase shall occur after those defined
       herein.
       The values of phaseNumber used here shall not
       exceed maxPhases or violate the Consistency Checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptDwellPhase
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 13 }
```

2.7.2.14 Preempt Dwell Ped Parameter

```
preemptDwellPed   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a phaseNumber (binary value) that specifies
       the pedestrian movement(s) to be served in
       the Preempt Dwell interval. The peds defined in
       prempt CyclingPed shall occur after those defined
       herein.
       The values of phaseNumber used here shall not
       exceed maxPhases or violate the Consistency Checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptDwellPed
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 14 }
```

2.7.2.15 Preempt Exit Phase Parameter

```
preemptExitPhase   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a phaseNumber (binary value) that shall be
       active following Preempt.
       The values of phaseNumber used here shall not
       exceed maxPhases or violate the Consistency Checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptExitPhase
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 15 }
```

2.7.2.16 Preempt State

```
preemptState   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                      notActive (2),
                      notActiveWithCall (3),
                      entryStarted (4),
                      trackService (5),
                      dwell (6),
                      linkActive (7),
                      exitStarted (8),
                      maxPresence (9) }
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Preempt State provides status on which
       state the associated preempt is in.  The states are
       as follows:
          other: preempt service is not specified in this
             standard.
          notActive: preempt input is not active, this
             preempt is not active.
          notActiveWithCall: preempt input is active,
             preempt service has not started.
          entryStarted: preempt service is timing the entry
             intervals.
          trackService: preempt service is timing the track
             intervals.
          dwell: preempt service is timing the dwell
             intervals.
          linkActive: preempt service is performing linked
             operation.
          exitStarted: preempt service is timing the exit
             intervals.
          **maxPresence**: preempt input has exceeded
             maxPresence time
       <DescriptiveName> NTCIP-1202::ASC.preemptState
       <DataConceptType> Data Element"
::= { preemptEntry 16}
```

2.7.2.17 Preempt Track Overlap Parameter

```
preemptTrackOverlap   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a overlapNumber (binary value) that shall
       be active during the Preempt Track Clear intervals.
       The values of overlapNumber used here shall not
       exceed maxOverlaps or violate the consistency checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptTrackOverlap
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 17 }
```

2.7.2.18 Preempt Dwell Overlap Parameter

```
preemptDwellOverlap   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a overlapNumber (binary value) that is
       allowed during the Preempt Dwell interval.
       The values of overlapNumber used here shall not
       exceed maxOverlaps or violate the consistency checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptDwellOverlap
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 18 }
```

2.7.2.19 Preempt Cycling Phase Parameter

```
preemptCyclingPhase   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a phaseNumber (binary value) that is
       allowed to cycle during the Preempt Dwell interval.
       The values of phaseNumber used here shall not
       exceed maxPhases or violate the Consistency Checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptCyclingPhase
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 19 }
```

2.7.2.20 Preempt Cycling Ped Parameter

```
preemptCyclingPed   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a phaseNumber (binary value) indicating a
       pedestrian movement that is allowed to cycle during
       the Preempt Dwell interval.
       The values of phaseNumber used here shall not
       exceed maxPhases or violate the consistency checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptCyclingPed
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 20 }
```

2.7.2.21 Preempt Cycling Overlap Parameter

```
preemptCyclingOverlap   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet within the octet string
       contains a overlapNumber (binary value) that is
       allowed to cycle during the Preempt Dwell interval.
       The values of overlapNumber used here shall not
       exceed maxOverlaps or violate the consistency checks
       defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.preemptCyclingOverlap
       <DataConceptType> Data Element"
   DEFVAL   { "" }
::= { preemptEntry 21 }
```

2.7.2.22 Preempt Enter Yellow Change Parameter

```
preemptEnterYellowChange   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Enter Yellow Change in tenth seconds (0-25.5
       sec). This parameter controls the yellow change
       timing for a normal Yellow Change signal terminated by a
       preempt initiated transition.  A preempt initiated
       transition shall not cause the termination of a Yellow
       Change prior to its display for the lesser of the phase’s
       Yellow Change time or this period.  CAUTION - if this
       value is zero, phase Yellow Change is terminated immediately.
       <DescriptiveName> NTCIP-1202::ASC.preemptEnterYellowChange
       <DataConceptType> Data Element
       <Unit> tenth second"
   DEFVAL {255}
::= { preemptEntry 22 }
```

2.7.2.23 Preempt Enter Red Clear Parameter

```
preemptEnterRedClear   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Enter Red Clear in tenth seconds (0-25.5
       sec). This parameter controls the red clearance
       timing for a normal Red Clear signal terminated by a
       preempt initiated transition.  A preempt initiated
       transition shall not cause the termination of a Red
       Clear prior to its display for the lesser of the phase’s
       Red Clear time or this period.  CAUTION - if this value is
       zero, phase Red Clear is terminated immediately.
       <DescriptiveName> NTCIP-1202::ASC.preemptEnterRedClear
       <DataConceptType> Data Element
       <Unit> tenth second"
   DEFVAL {255}
::= { preemptEntry 23 }
```

2.7.2.24 Preempt Track Yellow Change Parameter

```
preemptTrackYellowChange   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Track Clear Yellow Change time in tenth
       seconds (0-25.5 sec). The lesser of the phase’s
       Yellow Change time or this parameter controls the
       yellow timing for the track clearance movement. Track
       clear phase(s) are enabled in the preemptTrackPhase object.
       <DescriptiveName> NTCIP-1202::ASC.preemptTrackYellowChange
       <DataConceptType> Data Element
       <Unit> tenth second"
   DEFVAL {255}
::= { preemptEntry 24 }
```

2.7.2.25 Preempt Track Red Clear Parameter

```
preemptTrackRedClear   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Track Clear Red Clear time in tenth seconds
       (0-25.5 sec). The lesser of the phase’s Red Clear
       time or this parameter controls the Red Clear
       timing for the track clearance movement.  Track clear
       phase(s) are enabled in the preemptTrackPhase object.
       <DescriptiveName> NTCIP-1202::ASC.preemptTrackRedClear
       <DataConceptType> Data Element
       <Unit> tenth second"
   DEFVAL {255}
::= { preemptEntry 25 }
```

2.7.3 Preempt Control Table

```
preemptControlTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF PreemptControlEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> This table contains the control objects
       that allow the preempts to be activated remotely.
       There shall be one control object for each preempt
       input supported by the device.  The number of rows
       in this table shall be equal to maxPreempts.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.preemptControlTable
       <DataConceptType> Entity Type"
::= { preempt 3 }

preemptControlEntry   OBJECT-TYPE
   SYNTAX   PreemptControlEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Control objects for each preempt input.
       These objects allow the system to activate preempt
       functions remotely.
       <DescriptiveName> NTCIP-1202::ASC.preemptControlEntry
       <DataConceptType> Entity Type"
   INDEX   { preemptControlNumber }
::= { preemptControlTable 1 }

PreemptControlEntry ::= SEQUENCE {
   preemptControlNumber   INTEGER,
   preemptControlState    INTEGER }
```

2.7.3.1 Preempt Control Number

```
preemptControlNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object shall indicate the preempt
       input number controlled by the associated
       preemptControlState object in this row.
       <DescriptiveName> NTCIP-1202::ASC.preemptControlNumber
       <DataConceptType> Data Element
       <Unit> preempt"
::= { preemptControlEntry  1 }
```

2.7.3.2 Preempt Control State

```
preemptControlState   OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object when set to ON (one)
       shall cause the associated preempt actions to occur
       unless the actions have already been started by the
       physical preempt input. The preempt shall remain
       active as long as this object is ON or the physical
       preempt input is ON.  This object when set to OFF
       (zero) shall cause the physical preempt input to
       control the associated preempt actions.

       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.preemptControlState
       <DataConceptType> Data Element"
::= { preemptControlEntry  2 }
```

2.8 RING PARAMETERS

```
ring  OBJECT IDENTIFIER
::= { asc  7 }

-- The ring node contains objects that support ring configuration,
-- status and control functions in the device.
```

2.8.1 Maximum Rings

```
maxRings   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The value of this object shall specify
       the maximum number of rings this device supports.
       <DescriptiveName> NTCIP-1202::ASC.maxRings
       <DataConceptType> Data Element
       <Unit> ring"
::= { ring 1 }
```

2.8.2 Maximum Sequences

```
maxSequences   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The value of this object shall specify
       the maximum number of sequence plans this device
       supports.
       <DescriptiveName> NTCIP-1202::ASC.maxSequences
       <DataConceptType> Data Element
       <Unit> sequence"
::= { ring 2 }
```

2.8.3 Sequence Table

```
sequenceTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF SequenceEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> This table contains all the sequence
       plans for the controller. A sequence plan shall
       consist of one row for each ring that the CU
       supports.  Each row defines the phase service order
       for that ring.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.sequenceTable
       <DataConceptType> Entity Type"
::= { ring 3 }

sequenceEntry   OBJECT-TYPE
   SYNTAX   SequenceEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Phase Sequence Parameters for an
       Actuated Controller Unit.
       <DescriptiveName> NTCIP-1202::ASC.sequenceEntry
       <DataConceptType> Entity Type"
   INDEX   { sequenceNumber, sequenceRingNumber }
::= { sequenceTable 1 }

SequenceEntry ::= SEQUENCE {
   sequenceNumber       INTEGER,
   sequenceRingNumber   INTEGER,
   sequenceData         OCTET STRING }
```

2.8.3.1 Sequence Number

```
sequenceNumber  OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS  read-only
   STATUS  optional
   DESCRIPTION
      "<Definition> This number identifies a sequence plan.
       Each row of the table contains the phase sequence
       for a ring.  A sequence plan shall consist of one
       row for each ring that defines the phase sequences
       for that ring.
       <DescriptiveName> NTCIP-1202::ASC.sequenceNumber
       <DataConceptType> Data Element
       <Unit> sequence"
::= { sequenceEntry 1 }
```

2.8.3.2 Sequence Ring Number

```
sequenceRingNumber  OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS  read-only
   STATUS  optional
   DESCRIPTION
      "<Definition> This number identifies the ring number
       this phase sequence applies to.
       <DescriptiveName> NTCIP-1202::ASC.sequenceRingNumber
       <DataConceptType> Data Element
       <Unit> ring"
::= { sequenceEntry 2 }
```

2.8.3.3 Sequence Data

```
sequenceData   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet is a Phase Number (binary
       value) within the associated ring number.  The phase
       number value shall not exceed the maxPhases object
       value.  The order of phase numbers determines the
       phase sequence for the ring.  The phase numbers
       shall not be ordered in a manner that would violate
       the Consistency Checks defined in Annex B.
       <DescriptiveName> NTCIP-1202::ASC.sequenceData
       <DataConceptType> Data Element"
::= { sequenceEntry 3 }
```

2.8.4 Maximum Ring Control Groups

```
maxRingControlGroups   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The maximum number of Ring Control
       Groups (8 rings per group) this Actuated Controller
       Unit supports.  This value is equal to
       TRUNCATE[(maxRings + 7) / 8]. This object indicates
       the maximum rows which shall appear in the
       ringControlGroupTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxRingControlGroups
       <DataConceptType> Data Element
       <Unit> group"
::= { ring 4 }
```

2.8.5 Ring Control Group Table

```
ringControlGroupTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF RingControlGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit Ring Control in groups of eight rings. The
       number of rows in this table is equal to the
       maxRingControlGroups object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupTable
       <DataConceptType> Entity Type"
::= { ring 5 }

ringControlGroupEntry   OBJECT-TYPE
   SYNTAX   RingControlGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Ring Control for eight Actuated
       Controller Unit rings.
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupEntry
       <DataConceptType> Entity Type"
   INDEX   { ringControlGroupNumber }
::= { ringControlGroupTable 1 }

RingControlGroupEntry ::= SEQUENCE {
   ringControlGroupNumber         INTEGER,
   ringControlGroupStopTime       INTEGER,
   ringControlGroupForceOff       INTEGER,
   ringControlGroupMax2           INTEGER,
   ringControlGroupMaxInhibit     INTEGER,
   ringControlGroupPedRecycle     INTEGER,
   ringControlGroupRedRest        INTEGER,
   ringControlGroupOmitRedClear   INTEGER }
```

2.8.5.1 Ring Control Group Number

```
ringControlGroupNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Ring Control Group number for
       objects in this row. This value shall not exceed the
       maxRingControlGroups object value.
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupNumber
       <DataConceptType> Data Element
       <Unit> group"
::= { ringControlGroupEntry 1 }
```

2.8.5.2 Ring Stop Time Control

```
ringControlGroupStopTime   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to stop timing in the device. The device
       shall activate/deactivate the System Stop Time
       control for a ring according to the respective bit
       value as follows:
             bit = 0 - deactivate the ring control
             bit = 1 - activate the ring control
          Bit 7: Ring # = (ringControlGroupNumber * 8)
          Bit 6: Ring # = (ringControlGroupNumber * 8) - 1
          Bit 5: Ring # = (ringControlGroupNumber * 8) - 2
          Bit 4: Ring # = (ringControlGroupNumber * 8) - 3
          Bit 3: Ring # = (ringControlGroupNumber * 8) - 4
          Bit 2: Ring # = (ringControlGroupNumber * 8) - 5
          Bit 1: Ring # = (ringControlGroupNumber * 8) - 6
          Bit 0: Ring # = (ringControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupStopTime
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.4.1.6"
::= { ringControlGroupEntry 2 }
```

2.8.5.3 Ring Force Off Control

```
ringControlGroupForceOff   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to terminate phases via a force off command
       in the device. The device shall activate/deactivate
       the System Force Off control for a ring according
       to the respective bit value as follows:
             bit = 0 - deactivate the ring control
             bit = 1 - activate the ring control
          Bit 7: Ring # = (ringControlGroupNumber * 8)
          Bit 6: Ring # = (ringControlGroupNumber * 8) - 1
          Bit 5: Ring # = (ringControlGroupNumber * 8) - 2
          Bit 4: Ring # = (ringControlGroupNumber * 8) - 3
          Bit 3: Ring # = (ringControlGroupNumber * 8) - 4
          Bit 2: Ring # = (ringControlGroupNumber * 8) - 5
          Bit 1: Ring # = (ringControlGroupNumber * 8) - 6
          Bit 0: Ring # = (ringControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupForceOff
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.4.1.1"
::= { ringControlGroupEntry 3 }
```

2.8.5.4 Ring Max 2 Control

```
ringControlGroupMax2   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to request Maximum 2 timings in the device.
       The device shall activate/deactivate the System
       Maximum 2 control for a ring according to the
       respective bit value as follows:
             bit = 0 - deactivate the ring control
             bit = 1 - activate the ring control
          Bit 7: Ring # = (ringControlGroupNumber * 8)
          Bit 6: Ring # = (ringControlGroupNumber * 8) - 1
          Bit 5: Ring # = (ringControlGroupNumber * 8) - 2
          Bit 4: Ring # = (ringControlGroupNumber * 8) - 3
          Bit 3: Ring # = (ringControlGroupNumber * 8) - 4
          Bit 2: Ring # = (ringControlGroupNumber * 8) - 5
          Bit 1: Ring # = (ringControlGroupNumber * 8) - 6
          Bit 0: Ring # = (ringControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupMax2
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.4.1.7"
::= { ringControlGroupEntry 4 }
```

2.8.5.5 Ring Max Inhibit Control

```
ringControlGroupMaxInhibit   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to request internal maximum timings be
       inhibited in the device. The device shall
       activate/deactivate the System Max Inhibit control
       for a ring according to the respective bit value as
       follows:
             bit = 0 - deactivate the ring control
             bit = 1 - activate the ring control
          Bit 7: Ring # = (ringControlGroupNumber * 8)
          Bit 6: Ring # = (ringControlGroupNumber * 8) - 1
          Bit 5: Ring # = (ringControlGroupNumber * 8) - 2
          Bit 4: Ring # = (ringControlGroupNumber * 8) - 3
          Bit 3: Ring # = (ringControlGroupNumber * 8) - 4
          Bit 2: Ring # = (ringControlGroupNumber * 8) - 5
          Bit 1: Ring # = (ringControlGroupNumber * 8) - 6
          Bit 0: Ring # = (ringControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupMaxInhibit
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.4.1.3"
::= { ringControlGroupEntry 5 }
```

2.8.5.6 Ring Ped Recycle Control

```
ringControlGroupPedRecycle   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to request a pedestrian recycle in the
       device. The device shall activate/deactivate the
       System Ped Recycle control for a ring according to
       the respective bit value as follows:
             bit = 0 - deactivate the ring control
             bit = 1 - activate the ring control
          Bit 7: Ring # = (ringControlGroupNumber * 8)
          Bit 6: Ring # = (ringControlGroupNumber * 8) - 1
          Bit 5: Ring # = (ringControlGroupNumber * 8) - 2
          Bit 4: Ring # = (ringControlGroupNumber * 8) - 3
          Bit 3: Ring # = (ringControlGroupNumber * 8) - 4
          Bit 2: Ring # = (ringControlGroupNumber * 8) - 5
          Bit 1: Ring # = (ringControlGroupNumber * 8) - 6
          Bit 0: Ring # = (ringControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupPedRecycle
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.4.1.5"
::= { ringControlGroupEntry 6 }
```

2.8.5.7 Ring Red Rest Control

```
ringControlGroupRedRest   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to request red rest in the device. The
       device shall activate/deactivate the System Red
       Rest control for a ring according to the
       respective bit value as follows:
             bit = 0 - deactivate the ring control
             bit = 1 - activate the ring control
```

```
          Bit 7: Ring # = (ringControlGroupNumber * 8)
          Bit 6: Ring # = (ringControlGroupNumber * 8) - 1
          Bit 5: Ring # = (ringControlGroupNumber * 8) - 2
          Bit 4: Ring # = (ringControlGroupNumber * 8) - 3
          Bit 3: Ring # = (ringControlGroupNumber * 8) - 4
          Bit 2: Ring # = (ringControlGroupNumber * 8) - 5
          Bit 1: Ring # = (ringControlGroupNumber * 8) - 6
          Bit 0: Ring # = (ringControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupRedRest
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.4.1.2"
::= { ringControlGroupEntry 7 }
```

2.8.5.8 Ring Omit Red Control

```
ringControlGroupOmitRedClear   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to allow a remote
       entity to omit red clearances in the device. The
       device shall activate/deactivate the System Omit
       Red Clear control for a ring according to the
       respective bit value as follows:
             bit = 0 - deactivate the ring control
             bit = 1 - activate the ring control
          Bit 7: Ring # = (ringControlGroupNumber * 8)
          Bit 6: Ring # = (ringControlGroupNumber * 8) - 1
          Bit 5: Ring # = (ringControlGroupNumber * 8) - 2
          Bit 4: Ring # = (ringControlGroupNumber * 8) - 3
          Bit 3: Ring # = (ringControlGroupNumber * 8) - 4
          Bit 2: Ring # = (ringControlGroupNumber * 8) - 5
          Bit 1: Ring # = (ringControlGroupNumber * 8) - 6
          Bit 0: Ring # = (ringControlGroupNumber * 8) - 7
       The device shall reset this object to ZERO when in
       BACKUP Mode. A write to this object shall reset the
       Backup timer to ZERO (see unitBackupTime).
       <DescriptiveName> NTCIP-1202::ASC.ringControlGroupOmitRedClear
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.5.4.1.4"
::= { ringControlGroupEntry 8 }
```

2.8.6 Ring Status Table

```
ringStatusTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF RingStatusEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit Ring Status.The number of rows in this table is
       equal to the maxRings object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.ringStatusTable
       <DataConceptType> Entity Type"
::= { ring 6 }

ringStatusEntry   OBJECT-TYPE
   SYNTAX   RingStatusEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Ring Status for an Actuated Controller
       Unit ring.
       <DescriptiveName> NTCIP-1202::ASC.ringStatusEntry
       <DataConceptType> Entity Type"
   INDEX   { sequenceRingNumber }
::= { ringStatusTable 1 }

RingStatusEntry ::= SEQUENCE {
   ringStatus   INTEGER }
```

2.8.6.1 Ring Status

```
ringStatus   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Ring Status for this ring.
          Bit 7: Reserved (always zero)
          Bit 6: Reserved (always zero)
          Bit 5: Force Off - When bit = 1, the active
             phase in the ring was terminated by Force Off
          Bit 4: Max Out - When bit = 1, the active
             phase in the ring was terminated by Max Out
          Bit 3: Gap Out - When bit = 1, the active
             phase in the ring was terminated by Gap Out
          Bit 2: Coded Status Bit C
          Bit 1: Coded Status Bit B
          Bit 0: Coded Status Bit A
             +======+=====+=====+=====+===============+
             | Code |    Bit States   | State         |
             |  ##  |  A  |  B  |  C  | Names         |
             +======+=====+=====+=====+===============+
             |   0  |  0  |  0  |  0  | Min Green     |
             |   1  |  1  |  0  |  0  | Extension     |
             |   2  |  0  |  1  |  0  | Maximum       |
             |   3  |  1  |  1  |  0  | Green Rest    |
             |   4  |  0  |  0  |  1  | Yellow Change |
             |   5  |  1  |  0  |  1  | Red Clearance |
             |   6  |  0  |  1  |  1  | Red Rest      |
             |   7  |  1  |  1  |  1  | Undefined     |
             +======+=====+=====+=====+===============+
          NEMA TS 2 Clause 3.5.4.2 provides further
          definition of Coded Status Bits.
       <DescriptiveName> NTCIP-1202::ASC.ringStatus
       <DataConceptType> Data Element"
::= { ringStatusEntry 1 }
```

2.9 CHANNEL PARAMETERS

```
channel OBJECT IDENTIFIER
::= { asc 8 }

--This defines a node for supporting channel objects.
```

2.9.1 Maximum Channels

```
maxChannels   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Channels this
       Actuated Controller Unit supports. This object
       indicates the maximum rows which shall appear in the
       channelTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxChannels
       <DataConceptType> Data Element
       <Unit> channel"
::= { channel 1 }
```

2.9.2 Channel Table

```
channelTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF ChannelEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit channel parameters. The number of rows in this
       table is equal to the maxChannels object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.channelTable
       <DataConceptType> Entity Type"
::= { channel 2 }

channelEntry   OBJECT-TYPE
   SYNTAX   ChannelEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Parameters for a specific Actuated
       Controller Unit channel.
       <DescriptiveName> NTCIP-1202::ASC.channelEntry
       <DataConceptType> Entity Type"
   INDEX   { channelNumber }
::= { channelTable 1 }

ChannelEntry ::= SEQUENCE {
   channelNumber          INTEGER,
   channelControlSource   INTEGER,
   channelControlType     INTEGER,
   channelFlash           INTEGER,
   channelDim             INTEGER }
```

2.9.2.1 Channel Number

```
channelNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The channel number for objects in this
       row. This value shall not exceed the maxChannels
       object value.
       <DescriptiveName> NTCIP-1202::ASC.channelNumber
       <DataConceptType> Data Element
       <Unit> channel"
::= { channelEntry 1 }
```

2.9.2.2 Channel Control Source Parameter

```
channelControlSource   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the channel control
       source (which Phase or Overlap). The value shall not
       exceed maxPhases or maxOverlaps as determined by
       channelControlType object:
          Value 00 = No Control (Not In Use)
          Value 01 = Phase 01 or Overlap A
          Value 02 = Phase 02 or Overlap B
                        ||
          Value 15 = Phase 15 or Overlap O
          Value 16 = Phase 16 or Overlap P
                       etc.
       <DescriptiveName> NTCIP-1202::ASC.channelControlSource
       <DataConceptType> Data Element"
::= { channelEntry 2 }
```

2.9.2.3 Channel Control Type Parameter

```
channelControlType   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                      phaseVehicle (2),
                      phasePedestrian (3),
                      overlap (4) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the channel control
       type (Vehicle Phase, Pedestrian Phase, or Overlap):
         other: The channel controls an other type of
            display.
         phaseVehicle: The channel controls a vehicle phase
               display.
         phasePedestrian: The channel controls a pedestrian
               phase display.
         overlap: The channel controls an overlap display.
       <DescriptiveName> NTCIP-1202::ASC.channelControlType
       <DataConceptType> Data Element"
::= { channelEntry 3 }
```

2.9.2.4 Channel Flash Parameter

```
channelFlash   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the channel state
       during Automatic Flash.
          Bit 7: Reserved
          Bit 6: Reserved
          Bit 5: Reserved
          Bit 4: Reserved
          Bit 3: Flash Alternate Half Hertz
             Bit=0: Off/Disabled & Bit=1: On/Enabled
          Bit 2: Flash Red
             Bit=0: Off/Red Dark & Bit=1: On/Flash Red
          Bit 1: Flash Yellow
             Bit=0: Off/Yellow Dark & Bit=1: On/Flash Yellow
          Bit 0: Reserved
       A SET of both bits 1 & 2 shall result in bit 1=0 and
       bit 2=1.
       A SET of a 'reserved' bit to a value other than
       zero (0) shall return a badValue(3) error.
       <DescriptiveName> NTCIP-1202::ASC.channelFlash
       <DataConceptType> Data Element"
::= { channelEntry 4 }
```

2.9.2.5 Channel Dim Parameter

```
channelDim   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the channel state
       during Dimming. Dimming shall be accomplished by the
       elimination of alternate one-half segments from the
       AC sinusoid applied to the field terminals.
          Bit 7: Reserved
          Bit 6: Reserved
          Bit 5: Reserved
          Bit 4: Reserved
          Bit 3: Dim Alternate Half Line Cycle
             Bit=0: Off/+ half cycle &
             Bit=1: On/- half cycle
          Bit 2: Dim Red
             Bit=0: Off/Red Not Dimmed &
             Bit=1: On/Dimmed Red
          Bit 1: Dim Yellow
             Bit=0: Off / Yellow Not Dimmed &
             Bit=1: On / Dimmed Yellow
          Bit 0: Dim Green
             Bit=0: Off / Green Not Dimmed &
             Bit=1: On / Dimmed Green
       A SET of a 'reserved' bit to a value other than
       zero (0) shall return a badValue(3) error.
       <DescriptiveName> NTCIP-1202::ASC.channelDim
       <DataConceptType> Data Element"
::= { channelEntry 5 }
```

2.9.3 Maximum Channel Status Groups

```
maxChannelStatusGroups   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The maximum number of Channel Status
       Groups (8 channels per group) this Actuated
       Controller Unit supports. This value is equal to
       TRUNCATE [(maxChannels + 7) / 8]. This object
       indicates the maximum rows which shall appear in
       the channelStatusGroupTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxChannelStatusGroups
       <DataConceptType> Data Element
       <Unit> group"
::= { channel 3 }
```

2.9.4 Channel Status Group Table

```
channelStatusGroupTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF ChannelStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit channel output (Red, Yellow, & Green) status in
       groups of eight channels. The number of rows in this
       table is equal to the maxChannelStatusGroups object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.channelStatusGroupTable
       <DataConceptType> Entity Type"
::= { channel 4 }

channelStatusGroupEntry   OBJECT-TYPE
   SYNTAX   ChannelStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Red, Yellow, & Green Output Status for
       eight Actuated Controller Unit channels.
       <DescriptiveName> NTCIP-1202::ASC.channelStatusGroupEntry
       <DataConceptType> Entity Type"
   INDEX   { channelStatusGroupNumber }
::= { channelStatusGroupTable 1 }
```

```
ChannelStatusGroupEntry ::= SEQUENCE {
   channelStatusGroupNumber    INTEGER,
   channelStatusGroupReds      INTEGER,
   channelStatusGroupYellows   INTEGER,
   channelStatusGroupGreens    INTEGER }
```

2.9.4.1 Channel Status Group Number

```
channelStatusGroupNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The channelStatusGroup number for
       objects in this row. This value shall not exceed the
       maxChannelStatusGroups object value.
       <DescriptiveName> NTCIP-1202::ASC.channelStatusGroupNumber
       <DataConceptType> Data Element
       <Unit> group"
::= { channelStatusGroupEntry 1 }
```

2.9.4.2 Channel Status Group Reds

```
channelStatusGroupReds   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Channel Red Output Status Mask, when a
       bit=1, the Channel Red is currently active. When a
       bit=0, the Channel Red is NOT currently active.
          Bit 7: Channel # = (channelStatusGroupNumber * 8)
          Bit 6: Channel # = (channelStatusGroupNumber * 8) - 1
          Bit 5: Channel # = (channelStatusGroupNumber * 8) - 2
          Bit 4: Channel # = (channelStatusGroupNumber * 8) - 3
          Bit 3: Channel # = (channelStatusGroupNumber * 8) - 4
          Bit 2: Channel # = (channelStatusGroupNumber * 8) - 5
          Bit 1: Channel # = (channelStatusGroupNumber * 8) - 6
          Bit 0: Channel # = (channelStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.channelStatusGroupReds
       <DataConceptType> Data Element"
::= { channelStatusGroupEntry 2 }
```

2.9.4.3 Channel Status Group Yellows

```
channelStatusGroupYellows   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Channel Yellow Output Status Mask, when
       a bit=1, the Channel Yellow is currently active. When
       a bit=0, the Channel Yellow is NOT currently active.
          Bit 7: Channel # = (channelStatusGroupNumber * 8)
          Bit 6: Channel # = (channelStatusGroupNumber * 8) - 1
          Bit 5: Channel # = (channelStatusGroupNumber * 8) - 2
          Bit 4: Channel # = (channelStatusGroupNumber * 8) - 3
          Bit 3: Channel # = (channelStatusGroupNumber * 8) - 4
          Bit 2: Channel # = (channelStatusGroupNumber * 8) - 5
          Bit 1: Channel # = (channelStatusGroupNumber * 8) - 6
          Bit 0: Channel # = (channelStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.channelStatusGroupYellows
       <DataConceptType> Data Element"
::= { channelStatusGroupEntry 3 }
```

2.9.4.4 Channel Status Group Greens

```
channelStatusGroupGreens   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Channel Green Output Status Mask, when
       a bit=1, the Channel Green is currently active. When
       a bit=0, the Channel Green is NOT currently active.
          Bit 7: Channel # = (channelStatusGroupNumber * 8)
          Bit 6: Channel # = (channelStatusGroupNumber * 8) - 1
          Bit 5: Channel # = (channelStatusGroupNumber * 8) - 2
          Bit 4: Channel # = (channelStatusGroupNumber * 8) - 3
          Bit 3: Channel # = (channelStatusGroupNumber * 8) - 4
          Bit 2: Channel # = (channelStatusGroupNumber * 8) - 5
          Bit 1: Channel # = (channelStatusGroupNumber * 8) - 6
          Bit 0: Channel # = (channelStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.channelStatusGroupGreens
       <DataConceptType> Data Element"
::= { channelStatusGroupEntry 4 }
```

2.10 OVERLAP PARAMETERS

```
overlap   OBJECT IDENTIFIER
::= { asc 9 }

--"This node contains objects that configure, monitor and
-- control overlap functions."
```

2.10.1 Maximum Overlaps

```
maxOverlaps   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Overlaps this
       Actuated Controller Unit supports. This object
       indicates the maximum number of rows which shall
       appear in the overlapTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxOverlaps
       <DataConceptType> Data Element
       <Unit> overlap"
::= { overlap 1 }
```

2.10.2 Overlap Table

```
overlapTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF OverlapEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit overlap parameters. The number of rows in this
       table is equal to the maxOverlaps object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.overlapTable
       <DataConceptType> Entity Type"
::= { overlap 2 }

overlapEntry   OBJECT-TYPE
   SYNTAX   OverlapEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Parameters for a specific Actuated
       Controller Unit overlap.
       <DescriptiveName> NTCIP-1202::ASC.overlapEntry
       <DataConceptType> Entity Type"
   INDEX   { overlapNumber }
::= { overlapTable 1 }

OverlapEntry ::= SEQUENCE {
   overlapNumber           INTEGER,
   overlapType             INTEGER,
   overlapIncludedPhases   OCTET STRING,
   overlapModifierPhases   OCTET STRING,
   overlapTrailGreen       INTEGER,
   overlapTrailYellow      INTEGER,
   overlapTrailRed         INTEGER }
```

2.10.2.1 Overlap Number

```
overlapNumber   OBJECT-TYPE
   SYNTAX  INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The overlap number for objects in this
       row. The value shall not exceed the maxOverlaps
       object. The value maps to the Overlap as follows:
       1 = Overlap A, 2 = Overlap B etc.
       <DescriptiveName> NTCIP-1202::ASC.overlapNumber
       <DataConceptType> Data Element
       <Unit> overlap"
::= { overlapEntry 1 }
```

2.10.2.2 Overlap Type

```
overlapType   OBJECT-TYPE
   SYNTAX   INTEGER { other(1),
                      normal (2),
                      minusGreenYellow (3) }
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> The type of overlap operation for this
       row.  The types are as follows:
         other: The overlap operates in another mode than
            those described herein.
         normal: The overlap output shall be controlled by
            the overlapIncludedPhases when this type is
            indicated. The overlap output shall be green in
            the following situations:
               (1) when an overlap included phase is green.
               (2) when an overlap included phase is yellow
                   (or red clearance) and an overlap
                   included phase is next.

            The overlap output shall be yellow when an
            included phase is yellow and an overlap included
            phase is not next. The overlap output shall be
            red whenever the overlap green and yellow are
            not ON.
         minusGreenYellow: The overlap output shall be
            controlled by the overlapIncludedPhases and the
            overlapModifierPhases if this type is indicated.
            The overlap output shall be green in the
            following situations:
              (1) when an overlap included phase is green
                  and an overlap modifier phase is NOT green.
              (2) when an overlap included phase is yellow
                  (or red clearance) and an overlap included
                  phase is next and an overlap modifier
                  phase is NOT green.

            The overlap output shall be yellow when an
            overlap included phase is yellow and an overlap
            modifier phase is NOT yellow and an overlap
            included phase is not next. The overlap output
            shall be red whenever the overlap green and
            yellow are not ON.
       <DescriptiveName> NTCIP-1202::ASC.overlapType
       <DataConceptType> Data Element"
::= { overlapEntry 2 }
```

2.10.2.3 Overlap Included Phase Parameter

```
overlapIncludedPhases   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet is a Phase (number) that
       shall be an included phase for the overlap. The
       phase number value shall not exceed the maxPhases
       object value. When an included phase output is green
       or when the CU is cycling between included phases,
       the overlap output shall be green.
       <DescriptiveName> NTCIP-1202::ASC.overlapIncludedPhases
       <DataConceptType> Data Element"
::= { overlapEntry 3 }
```

2.10.2.4 Overlap Modifier Phase Parameter

```
overlapModifierPhases   OBJECT-TYPE
   SYNTAX   OCTET STRING
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Each octet is a Phase (number) that
       shall be a modifier phase for the overlap. The phase
       number value shall not exceed the maxPhases object
       value.

       A null value provides a normal overlap type. A
       non-null value provides a minusGreenYellow overlap
       type.
       <DescriptiveName> NTCIP-1202::ASC.overlapModifierPhases
       <DataConceptType> Data Element"
::= { overlapEntry 4 }
```

2.10.2.5 Overlap Trailing Green Parameter

```
overlapTrailGreen   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Overlap Trailing Green Parameter in
       seconds (0-255 sec). When this value is greater than
       zero and the overlap green would normally terminate,
       the overlap green shall be extended by this
       additional time.
       <DescriptiveName> NTCIP-1202::ASC.overlapTrailGreen
       <DataConceptType> Data Element
       <Unit> second"
::= { overlapEntry 5 }
```

2.10.2.6 Overlap Trailing Yellow Change Parameter

```
overlapTrailYellow   OBJECT-TYPE
   SYNTAX  INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Overlap Trailing Yellow Change Parameter
       in tenth seconds (NEMA range: 3.0-25.5 sec). When
       the overlap green has been extended (Trailing Green),
       this value shall determine the current length of the
       Yellow Change interval for the overlap.
       <DescriptiveName> NTCIP-1202::ASC.overlapTrailYellow
       <DataConceptType> Data Element
       <Unit> tenth second"
::= { overlapEntry 6 }
```

2.10.2.7 Overlap Trailing Red Clear Parameter

```
overlapTrailRed   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> Overlap Trailing Red Clear Parameter in
       tenth seconds (0-25.5 sec). When the overlap green
       has been extended (Trailing Green), this value shall
       determine the current length of the Red Clearance
       interval for the overlap.
       <DescriptiveName> NTCIP-1202::ASC.overlapTrailRed
       <DataConceptType> Data Element
       <Unit> tenth second"
::= { overlapEntry 7 }
```

2.10.3 Maximum Overlap Status Groups

```
maxOverlapStatusGroups   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Overlap Status
       Groups (8 overlaps per group) this Actuated
       Controller Unit supports. This value is equal to
       TRUNCATE [(maxOverlaps + 7) / 8]. This object
       indicates the maximum rows which shall appear in the
       overlapStatusGroupTable object.
       <DescriptiveName> NTCIP-1202::ASC.maxOverlapStatusGroups
       <DataConceptType> Data Element
       <Unit> group"
::= { overlap 3 }
```

2.10.4 Overlap Status Group Table

```
overlapStatusGroupTable   OBJECT-TYPE
   SYNTAX   SEQUENCE OF OverlapStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit overlap output (Red, Yellow, & Green) status in
       groups of eight overlaps. The number of rows in this
       table is equal to the maxOverlapStatusGroups object.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.overlapStatusGroupTable
       <DataConceptType> Entity Type"
::= { overlap 4 }

overlapStatusGroupEntry   OBJECT-TYPE
   SYNTAX   OverlapStatusGroupEntry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> Red, Yellow, & Green Output Status for
       eight Actuated Controller Unit overlaps.
       <DescriptiveName> NTCIP-1202::ASC.overlapStatusGroupEntry
       <DataConceptType> Entity Type"
   INDEX   { overlapStatusGroupNumber }
::= { overlapStatusGroupTable 1 }

OverlapStatusGroupEntry ::= SEQUENCE {
   overlapStatusGroupNumber    INTEGER,
   overlapStatusGroupReds      INTEGER,
   overlapStatusGroupYellows   INTEGER,
   overlapStatusGroupGreens    INTEGER }
```

2.10.4.1 Overlap Status Group Number

```
overlapStatusGroupNumber   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The overlap StatusGroup number for
       objects in this row. This value shall not exceed the
       maxOverlapStatusGroups object value.
       <DescriptiveName> NTCIP-1202::ASC.overlapStatusGroupNumber
       <DataConceptType> Data Element
       <Unit> group"
::= { overlapStatusGroupEntry 1 }
```

2.10.4.2 Overlap Status Group Reds

```
overlapStatusGroupReds   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Overlap Red Output Status Mask, when a
       bit=1, the Overlap Red is currently active. When a
       bit=0, the Overlap Red is NOT currently active.
          Bit 7: Overlap # = (overlapStatusGroupNumber * 8)
          Bit 6: Overlap # = (overlapStatusGroupNumber * 8) - 1
          Bit 5: Overlap # = (overlapStatusGroupNumber * 8) - 2
          Bit 4: Overlap # = (overlapStatusGroupNumber * 8) - 3
          Bit 3: Overlap # = (overlapStatusGroupNumber * 8) - 4
          Bit 2: Overlap # = (overlapStatusGroupNumber * 8) - 5
          Bit 1: Overlap # = (overlapStatusGroupNumber * 8) - 6
          Bit 0: Overlap # = (overlapStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.overlapStatusGroupReds
       <DataConceptType> Data Element"
::= { overlapStatusGroupEntry 2 }
```

2.10.4.3 Overlap Status Group Yellows

```
overlapStatusGroupYellows   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Overlap Yellow Output Status Mask, when
       a bit=1, the Overlap Yellow is currently active. When
       a bit=0, the Overlap Yellow is NOT currently active.
          Bit 7: Overlap # = (overlapStatusGroupNumber * 8)
          Bit 6: Overlap # = (overlapStatusGroupNumber * 8) - 1
          Bit 5: Overlap # = (overlapStatusGroupNumber * 8) - 2
          Bit 4: Overlap # = (overlapStatusGroupNumber * 8) - 3
          Bit 3: Overlap # = (overlapStatusGroupNumber * 8) - 4
          Bit 2: Overlap # = (overlapStatusGroupNumber * 8) - 5
          Bit 1: Overlap # = (overlapStatusGroupNumber * 8) - 6
          Bit 0: Overlap # = (overlapStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.overlapStatusGroupYellows
       <DataConceptType> Data Element"
::= { overlapStatusGroupEntry 3 }
```

2.10.4.4 Overlap Status Group Greens

```
overlapStatusGroupGreens   OBJECT-TYPE
   SYNTAX   INTEGER (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> Overlap Green Output Status Mask, when
       a bit=1, the Overlap Green is currently active. When
       a bit=0, the Overlap Green is NOT currently active.
          Bit 7: Overlap # = (overlapStatusGroupNumber * 8)
          Bit 6: Overlap # = (overlapStatusGroupNumber * 8) - 1
          Bit 5: Overlap # = (overlapStatusGroupNumber * 8) - 2
          Bit 4: Overlap # = (overlapStatusGroupNumber * 8) - 3
          Bit 3: Overlap # = (overlapStatusGroupNumber * 8) - 4
          Bit 2: Overlap # = (overlapStatusGroupNumber * 8) - 5
          Bit 1: Overlap # = (overlapStatusGroupNumber * 8) - 6
          Bit 0: Overlap # = (overlapStatusGroupNumber * 8) - 7
       <DescriptiveName> NTCIP-1202::ASC.overlapStatusGroupGreens
       <DataConceptType> Data Element"
::= { overlapStatusGroupEntry 4 }
```

2.11 TS2 PORT 1 PARAMETERS

```
ts2port1  OBJECT IDENTIFIER
::= { asc 10 }

-- This object is an identifier used to group all objects for
-- support of NEMA TS 2 (Clause 3.3.1) Port 1 activities.
```

2.11.1 Maximum Port 1 Addresses

```
maxPort1Addresses   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The Maximum Number of Port 1 addresses
       this Actuated Controller Unit supports. This object
       indicates the maximum rows which shall appear in the
       port1Table object.
       <DescriptiveName> NTCIP-1202::ASC.maxPort1Addresses
       <DataConceptType> Data Element
       <Unit> address"
::= { ts2port1 1 }
```

2.11.2 Port 1 Table

```
port1Table   OBJECT-TYPE
   SYNTAX   SEQUENCE OF Port1Entry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> A table containing Actuated Controller
       Unit port 1 parameters. The number of rows in this
       table is equal to maxPort1Addresses object. Address
       255 is reserved for the all stations (link devices)
       address.
       <TableType> static
       <DescriptiveName> NTCIP-1202::ASC.port1Table
       <DataConceptType> Entity Type"
::= { ts2port1 2 }
```

```
port1Entry   OBJECT-TYPE
   SYNTAX   Port1Entry
   ACCESS   not-accessible
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines a conceptual row in
       the port 1 Table.
       <DescriptiveName> NTCIP-1202::ASC.port1Entry
       <DataConceptType> Entity Type"
   INDEX { port1Number }
::= { port1Table 1 }

Port1Entry ::= SEQUENCE {
   port1Number          INTEGER,
   port1DevicePresent   INTEGER,
   port1Frame40Enable   INTEGER,
   port1Status          INTEGER,
   port1FaultFrame      INTEGER }
```

2.11.2.1 Port 1 Number

```
port1Number   OBJECT-TYPE
   SYNTAX   INTEGER (1..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> The (Port 1 address plus one) for
       objects in this row. This value shall not exceed the
       maxPort1Addresses object value.
       <DescriptiveName> NTCIP-1202::ASC.port1Number
       <DataConceptType> Data Element
       <Unit> address"
::= { port1Entry 1 }
```

2.11.2.2 Port 1 Device Present

```
port1DevicePresent   OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> This object is used to program the CU as to
       the presence or absence of a device for this Port 1 address.
       The CU shall transmit Command Frames only to those devices
       that are present as determined by this programming.
          True (one) - the device is present.
          False (zero) - the device is not present.
       <DescriptiveName> NTCIP-1202::ASC.port1DevicePresent
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.3.1.4"
::= { port1Entry 2 }
```

2.22.2.3 Port 1 Frame 40 Enable

```
port1Frame40Enable   OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> To enable or disable Frame 40 messages
       to the device at this Port 1 address. Frame 40 is
       used to poll the secondary stations for a secondary
       to secondary message exchange. Command 40 series
       frames shall be transmitted only to those devices
       that are enabled, as determined by this programming.
          TRUE (one) - Enable frame 40 messages for
             this device.
          FALSE (zero) - Disable frame 40 messages for this
             device.
       <DescriptiveName> NTCIP-1202::ASC.port1Frame40Enable
       <DataConceptType> Data Element"
   REFERENCE
      "NEMA TS 2 Clause 3.3.1.4.1"
::= { port1Entry 3 }
```

2.11.2.4 Port 1 Status

```
port1Status   OBJECT-TYPE
   SYNTAX   INTEGER { other (1),
                      online (2),
                      responseFault (3) }
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object indicates the
       communications status with the associated device:
          other: This indicates that some other
             communications faulthas been detected.
          online: This indicates that at least five of the
             most recent 10 response transfers were
             received correctly.
          responseFault: This indicates that more than 5 of
             the most recent 10 response transfers were
             received incorrectly.
       <DescriptiveName> NTCIP-1202::ASC.port1Status
       <DataConceptType> Data Element"
::= { port1Entry 4 }
```

2.11.2.5 Port 1 Fault Frame

```
port1FaultFrame   OBJECT-TYPE
   SYNTAX   INTEGER  (0..255)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object indicates the frame number
       that caused the most recent fault.
       <DescriptiveName> NTCIP-1202::ASC.port1FaultFrame
       <DataConceptType> Data Element"
::= { port1Entry 5 }
```

2.12 ASC BLOCK OBJECTS

```
ascBlock  OBJECT IDENTIFIER
::= { asc 11 }

-- This object is an identifier used to group all objects for
-- support of ASC Block Upload and Download activities.
```

2.12.1 ASC Block Get Control

```
ascBlockGetControl   OBJECT-TYPE
   SYNTAX   OCTET STRING (SIZE(2..12))
   ACCESS   read-write
   STATUS   optional
   DESCRIPTION
      "<Definition> An OER encoded string of reference
       parameters for ASC Block Uploads. The parameter
       values in this string are:
         ascBlockDataType    INTEGER (0..255)
         ascBlockDataID      INTEGER (0..255)
         ascBlockIndex1      INTEGER (0..255) if needed
         ascBlockQuantity1   INTEGER (0..255) if needed
         ascBlockIndex2      INTEGER (0..255) if needed
         ascBlockQuantity2   INTEGER (0..255) if needed
         ascBlockIndex3      INTEGER (0..255) if needed
         ascBlockQuantity3   INTEGER (0..255) if needed
         ascBlockIndex4      INTEGER (0..255) if needed
         ascBlockQuantity4   INTEGER (0..255) if needed
         ascBlockIndex5      INTEGER (0..255) if needed
         ascBlockQuantity5   INTEGER (0..255) if needed

       A GET of ascBlockData shall utilize values currently
       in this object to define the data to be returned.

       A SET of this object shall be evaluated for validity
       and Error Status of badValue(3) be returned for the
       following conditions:
        1) ascBlockDataType is not supported
        2) ascBlockDataID is not supported
        3) ascBlockIndex1 is zero or not supported
        4) ascBlockQuantity1 is zero or
           ascBlockIndex1 + ascBlockQuantity1 - 1 is not
           supported
        5) ascBlockIndex2 is zero or not supported
        6) ascBlockQuantity2 is zero or
           ascBlockIndex2 + ascBlockQuantity2) - 1 is not
           supported
        7) ascBlockIndex3 is zero or not supported
        8) ascBlockQuantity3 is zero or
           ascBlockIndex3 + ascBlockQuantity3) - 1 is not
           supported
        9) ascBlockIndex4 is zero or not supported
       10) ascBlockQuantity4 is zero or
           ascBlockIndex4 + ascBlockQuantity4) - 1 is not
           supported
       11) ascBlockIndex5 is zero or not supported
       12) ascBlockQuantity5 is zero or
           ascBlockIndex5 + ascBlockQuantity5) - 1 is not
           supported
       13) if the SET length is zero or incorrect for
           ascBlockDataType & ascBlockDataID
       14) if the GetResponse length for a GET on
           ascBlockData using maximum data field sizes
           would exceed a local limitation
       When this validity check fails, ascBlockErrorStatus
       shall be set equal to the Bullet Value above that
       generated the error.
       <DescriptiveName> NTCIP-1202::ASC.ascBlockGetControl
       <DataConceptType> Data Frame
       <Unit> "
::= { ascBlock 1 }
```

2.12.2 ASC Block Data

```
ascBlockData   OBJECT-TYPE
   SYNTAX   OCTET STRING (SIZE(2..484))
   ACCESS  read-write
   STATUS  optional
   DESCRIPTION
      "<Definition> An OER encoded string used for
       uploading and downloading ASC parameters. See
       SECTION 3 for encoding and decoding the block.

       A SET on this object shall require the use of
       'dbCreateTransaction' defined in NTCIP 1201
       Clause 2.3.1.

       A SET of this object shall be evaluated for validity
       and Error Status of badValue(3) be returned for the
       following conditions:
        1) ascBlockDataType is not supported
        2) ascBlockDataID is not supported
        3) ascBlockIndex1 is zero or not supported
        4) ascBlockQuantity1 is zero or
           ascBlockIndex1 + ascBlockQuantity1 - 1 is not
           supported
        5) ascBlockIndex2 is zero or not supported
        6) ascBlockQuantity2 is zero or
           ascBlockIndex2 + ascBlockQuantity2) - 1 is not
           supported
        7) ascBlockIndex3 is zero or not supported
        8) ascBlockQuantity3 is zero or
           ascBlockIndex3 + ascBlockQuantity3) - 1 is not
           supported
        9) ascBlockIndex4 is zero or not supported
       10) ascBlockQuantity4 is zero or
           ascBlockIndex4 + ascBlockQuantity4) - 1 is not
           supported
       11) ascBlockIndex5 is zero or not supported
       12) ascBlockQuantity5 is zero or
           ascBlockIndex5 + ascBlockQuantity5) - 1 is not
```

```
            supported
        13) if the SET length is zero or incorrect for
            ascBlockDataType & ascBlockDataID
        14) if the SET (SEQUENCE OF) value is incorrect.
        When this validity check fails, ascBlockErrorStatus
        shall be set equal to the Bullet Value above that
        generated the error.

        A SET that includes an unsupported value for a
        supported data element shall return an Error Status
        of badValue(3) and ascBlockErrorStatus shall be set
        equal to:
           (data Sequence # * 100) + data Element #

        A SET that includes a non-zero or non-null value in
        the position of an unsupported data element shall
        return an Error Status of badValue(3) and
        ascBlockErrorStatus shall be set equal to:
           (data Sequence # * 100) + data Element #

        A GET on this object shall utilize values currently in
        ascBlockGetControl to define the data to be returned.
        When ascBlockGetControl has invalid data, an Error
        Status of badValue(3) shall be returned.

        A GET shall return a zero or null value in the position
        of an unsupported object.
        <DescriptiveName> NTCIP-1202::ASC.ascBlockData
        <DataConceptType> Data Frame
        <Unit> "
::= { ascBlock 2 }
```

2.12.3 ASC Block Error Status

```
ascBlockErrorStatus   OBJECT-TYPE
   SYNTAX   INTEGER (0..65535)
   ACCESS   read-only
   STATUS   optional
   DESCRIPTION
      "<Definition> This object defines the data element
       within ascBlockGetControl or ascBlockData that
       caused a badValue(3) ErrorStatus.
         This object should equal zero after any successful
       SET to ascBlockGetControl or ascBlockData.
       <DescriptiveName> NTCIP-1202::ASC.ascBlockErrorStatus
       <DataConceptType> Data Element"
::= { ascBlock 3 }

END
```

Section 3
BLOCK OBJECT DEFINITIONS

3.1 BLOCK DATA TYPE & ID

All ASC Block Objects shall begin with two octets that define the Data Type and Data ID.

The Data Type octet (ascBlockDataType) provides for the definition of both NTCIP Standard and Device Proprietary data blocks. NTCIP Standard Data Blocks shall utilize an 'ascBlockDataType' of zero. Device Proprietary Data Blocks shall utilize an 'ascBlockDataType' equal to the Private Node Number (PNN) as assigned by NEMA (1.3.6.1.4.1.1206.3.PNN).

dataType	Description
0x00	Standard Data Block
0XPNN	Device Proprietary Data Block

The Data ID octet (ascBlockDataID) provides for definition of included data parameters. NCTIP Standard Data Blocks shall include an 'ascBlockDataID' as listed below:

ascBlockData-dataID Definitions		
dataID	Name	Description
0x00	AscPhaseBlock	Phase Data (see 3.2)
0x01	AscVehDetectorBlock	Vehicle Detector Data (see 3.3)
0x02	AscPedDetectorBlock	Pedestrian Detector Data (see 3.4)
0x03	AscPatternBlock	Pattern Data (see 3.5)
0x04	AscSplitBlock	Split Data (see 3.6)
0x05	AscTimebaseBlock	Time Base Data (see 3.7)
0x06	AscPreemptBlock	Preempt Data (see 3.8)
0x07	AscSequenceBlock	Sequence Data (see 3.9)
0x08	AscChannelBlock	Channel Data (see 3.10)
0x09	AscOverlapBlock	Overlap Data (see 3.11)
0x0A	AscPort1Block	Port 1 Data (see 3.12)
0x0B	AscScheduleBlock	Schedule Data (see 3.13)
0x0C	AscDayPlanBlock	Day Plan Data (see 3.14)
0x0D	AscEventConfigBlock	Event Config Data (see 3.15)
0x0E	AscEventClassBlock	Event Class Data (see 3.16)
0x0F	AscDynObjConfigBlock **(*)**	Dynamic Obj Config Data (see 3.17)
0x10	AscDynObjOwnerBlock **(*)**	Dynamic Obj Owner Data (see 3.18)
0x11	AscDynObjStatusBlock **(*)**	Dynamic Obj Status Data (see 3.19)
0x12	AscMiscBlock	Miscellaneous ASC Data (see 3.20)
0x13-0xFF		Reserved For NTCIP ASC Usage

(*) Any attempt to GET or SET this data via STMP shall result in a genError

New versions of this Standard shall NOT change the structure (content or definition) for any dataID block. New dataID blocks may be added for ascBlockData for expansion to cover other parameters. When a dataID block needs to be revised, the standard writers shall deprecate ascBlockData and establish a new OID (i.e., ascBlockData1) for all the current dataID blocks.

Proprietary Device Blocks shall include an 'ascBlockDataID' as defined in their separate documentation

3.2 PHASE BLOCK DATA

```
-- ascBlockData values for standard Block
-- Phase Data shall be as follows:

AscPhaseBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x00 phase data
    ascBlockIndex1       INTEGER (0..255),  -- phaseNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of phases

    -- for {
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscPhaseBlockData
   }

AscPhaseBlockData ::= SEQUENCE
   {
    phaseWalk.x                  INTEGER (0..255),
    phasePedestrianClear.x       INTEGER (0..255),
    phaseMinimumGreen.x          INTEGER (0..255),
    phasePassage.x               INTEGER (0..255),
    phaseMaximum1.x              INTEGER (0..255),
    phaseMaximum2.x              INTEGER (0..255),
    phaseYellowChange.x          INTEGER (0..255),
    phaseRedClear.x              INTEGER (0..255),
    phaseRedRevert.x             INTEGER (0..255),
    phaseAddedInitial.x          INTEGER (0..255),
    phaseMaximumInitial.x        INTEGER (0..255),
    phaseTimeBeforeReduction.x   INTEGER (0..255),
    phaseCarsBeforeReduction.x   INTEGER (0..255),
    phaseTimeToReduce.x          INTEGER (0..255),
    phaseReduceBy.x              INTEGER (0..255),
    phaseMinimumGap.x            INTEGER (0..255),
    phaseDynamicMaxLimit.x       INTEGER (0..255),
    phaseDynamicMaxStep.x        INTEGER (0..255),
    phaseStartup.x               INTEGER (1..6),
    phaseOptions.x               INTEGER (0..65535),
    phaseRing.x                  INTEGER (0..255),
    phaseConcurrency.x           OCTET STRING
   }
```

3.2.1 Phase Block Example

```
-- The following provides an example octet string value for
-- a set or get of a phase block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 00                ascBlockDataID (phase data)
-- 02                ascBlockIndex1 (start with phaseNumber=2)
-- 02                ascBlockQuantity1 (## of phases=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (phaseNumber=2)
-- 06                phaseWalk.2            (6 sec)
-- 0C                phasePedestrianClear.2 (12 sec)
--  |                etc, etc, to:
-- 01                phaseRing.2            (ring 1)
-- 02 05 06          phaseConcurrency.2     (ph 5 & 6)
--                SEQUENCE # 2 (phaseNumber=3)
-- 00                phaseWalk.3            (0 sec)
-- 00                phasePedestrianClear.3 (0 sec)
--  |                etc, etc, to:
-- 01                phaseRing.3            (ring 1)
-- 02 07 08          phaseConcurrency.3     (ph 7 & 8)
```

3.3 VEHICLE DETECTOR BLOCK DATA

```
-- ascBlockData values for standard Block
-- Vehicle Detector Data shall be as follows:

AscVehDetectorBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x01 veh detector data
    ascBlockIndex1       INTEGER (0..255),  -- vehicleDetectorNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of veh detectors

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscVehDetectorBlockData
   }

AscVehDetectorBlockData ::= SEQUENCE
   {
    vehicleDetectorOptions.x         INTEGER (0..255),
    vehicleDetectorCallPhase.x       INTEGER (0..255),
    vehicleDetectorSwitchPhase.x     INTEGER (0..255),
    vehicleDetectorDelay.x           INTEGER (0..65535),
    vehicleDetectorExtend.x          INTEGER (0..255),
    vehicleDetectorQueueLimit.x      INTEGER (0..255),
    vehicleDetectorNoActivity.x      INTEGER (0..255),
    vehicleDetectorMaxPresence.x     INTEGER (0..255),
    vehicleDetectorErraticCounts.x   INTEGER (0..255),
    vehicleDetectorFailTime.x        INTEGER (0..255)
   }
```

3.3.1 Vehicle Detector Block Example

```
-- The following provides an example octet string value for
-- a set or get of a vehicle detector block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 01                ascBlockDataID (veh detector data)
-- 02                ascBlockIndex1 (start with vehicleDetectorNumber=2)
-- 02                ascBlockQuantity1 (## of veh det=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (vehicleDetectorNumber =2)
-- B4                vehicleDetectorOptions.2       (bits)
-- 02                vehicleDetectorCallPhase.2     (ph 2)
--  |                etc, etc, to:
-- 00                vehicleDetectorErraticCounts.2 (0 cpm)
-- FF                vehicleDetectorFailTime.2      (255 sec)
--                SEQUENCE # 2 (vehicleDetectorNumber =3)
-- B4                vehicleDetectorOptions.3       (bits)
-- 03                vehicleDetectorCallPhase.3     (ph 3)
--  |                etc, etc, to:
-- 00                vehicleDetectorErraticCounts.3 (0 cpm)
-- FF                vehicleDetectorFailTime.3      (255 sec)
```

3.4 PEDESTRIAN DETECTOR BLOCK DATA

```
-- ascBlockData values for standard Block
-- Pedestrian Detector Data shall be as follows:

AscPedDetectorBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x02 ped detector data
    ascBlockIndex1       INTEGER (0..255),  -- pedestrianDetectorNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of ped detectors

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscPedDetectorBlockData
   }

AscPedDetectorBlockData ::= SEQUENCE
   {
    pedestrianDetectorCallPhase.x       INTEGER (0..255),
    pedestrianDetectorNoActivity.x      INTEGER (0..255),
    pedestrianDetectorMaxPresence.x     INTEGER (0..255),
    pedestrianDetectorErraticCounts.x   INTEGER (0..255)
   }
```

3.4.1 Pedestrian Detector Block Example

```
-- The following provides an example octet string value for
-- a set or get of a pedestrian detector block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 02                ascBlockDataID (ped detector data)
-- 02                ascBlockIndex1 (start with pedestrianDetectorNumber=2)
-- 02                ascBlockQuantity1 (## of ped det=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (pedestrianDetectorNumber =2)
-- 02                pedestrianDetectorCallPhase.2     (ph 2)
-- 00                pedestrianDetectorNoActivity.2    (0 min)
-- 00                pedestrianDetectorMaxPresence.2   (0 min)
-- 00                pedestrianDetectorErraticCounts.2 (0 cpm)
--                SEQUENCE # 2 (pedestrianDetectorNumber =3)
-- 03                pedestrianDetectorCallPhase.3     (ph 3)
-- 00                pedestrianDetectorNoActivity.3    (0 min)
-- 00                pedestrianDetectorMaxPresence.3   (0 min)
-- 00                pedestrianDetectorErraticCounts.3 (0 cpm)
```

3.5 PATTERN BLOCK DATA

```
-- ascBlockData values for standard Block
-- Pattern Data shall be as follows:

AscPatternBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x03 pattern data
    ascBlockIndex1       INTEGER (0..255),  -- patternNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of patterns

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscPatternBlockData
   }

AscPatternBlockData ::= SEQUENCE
   {
    patternCycleTime.x        INTEGER (0..255),
    patternOffsetTime.x       INTEGER (0..255),
    patternSequenceNumber.x   INTEGER (0..255)
   }
```

3.5.1 Pattern Block Example

```
-- The following provides an example octet string value for
-- a set or get of a pattern block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 03                ascBlockDataID (pattern data)
-- 02                ascBlockIndex1 (start with patternNumber=2)
-- 02                ascBlockQuantity1 (## of patterns=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (patternNumber =2)
-- 50                patternCycleTime.2      (80 sec)
-- 00                patternOffsetTime.2     (0 sec)
-- 01                patternSequenceNumber.2 (seq 1)
--                SEQUENCE # 2 (patternNumber =3)
-- 64                patternCycleTime.3      (100 sec)
-- 05                patternOffsetTime.3     (5 sec)
-- 01                patternSequenceNumber.3 (seq 1)
```

3.6 SPLIT BLOCK DATA

```
-- ascBlockData values for standard Block
-- Split Data shall be as follows:

AscSplitBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x04 split data
    ascBlockIndex1       INTEGER (0..255),  -- splitPhase
    ascBlockQuantity1    INTEGER (0..255),  -- ## of phases
    ascBlockIndex2       INTEGER (0..255),  -- splitNumber
    ascBlockQuantity2    INTEGER (0..255),  -- ## of splits

    -- for (
    --      y = ascBlockIndex2;
    --      y < (ascBlockIndex2 + ascBlockQuantity2);
    --      y++)
    --      for (
    --           x = ascBlockIndex1;
    --           x < (ascBlockIndex1 + ascBlockQuantity1);
    --           x++)

    data    SEQUENCE OF AscSplitBlockData
   }

AscSplitBlockData ::= SEQUENCE
   {
   splitTime.y.x         INTEGER (0..255),
   splitMode.y.x         INTEGER (1..7),
   splitCoordPhase.y.x   INTEGER (0..1)
  }
```

3.6.1 Split Block Example

```
-- The following provides an example octet string value for
-- a set or get of a split block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 04                ascBlockDataID (split data)
-- 01                ascBlockIndex1 (start with splitPhase=1)
-- 02                ascBlockQuantity1 (## of phases=2)
-- 01                ascBlockIndex2 (start with splitNumber=1)
-- 02                ascBlockQuantity2 (## of splits=2)
--                SEQUENCE OF
-- 01 04             quantity of items (ascBlockQuantity1 * ascBlockQuantity2)
--                SEQUENCE # 1 (splitNumber=1 / splitPhase=1)
-- 14                splitTime.1.1       (20 sec)
-- 02                splitMode.1.1       (none)
-- 00                splitCoordPhase.1.1 (false)
--                SEQUENCE # 2 (splitNumber=1 / splitPhase=2)
-- 14                splitTime.1.2       (20 sec)
-- 02                splitMode.1.2       (none)
-- 01                splitCoordPhase.1.2 (true)
--                SEQUENCE # 3 (splitNumber=2 / splitPhase=1)
-- 19                splitTime.2.1       (25 sec)
-- 02                splitMode.2.1       (none)
-- 00                splitCoordPhase.2.1 (false)
--                SEQUENCE # 4 (splitNumber=2 / splitPhase=2)
-- 19                splitTime.2.2       (25 sec)
-- 02                splitMode.2.2       (none)
-- 01                splitCoordPhase.2.2 (true)
```

3.7 TIME BASE BLOCK DATA

```
-- ascBlockData values for standard Block
-- Time Base Data shall be as follows:

AscTimebaseBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x05 time base data
    ascBlockIndex1       INTEGER (0..255),  -- timebaseAscActionNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of actions

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscTimebaseBlockData
   }

AscTimebaseBlockData ::= SEQUENCE
   {
    timebaseAscPattern.x             INTEGER (0..255),
    timebaseAscAuxillaryFunction.x   INTEGER (0..255),
    timebaseAscSpecialFunction.x     INTEGER (0..255)
   }
```

3.7.1 Time Base Block Example

```
-- The following provides an example octet string value for
-- a set or get of a time base block.
--
--                  SEQUENCE
-- 00                  ascBlockDataType (standard block)
-- 05                  ascBlockDataID (time base data)
-- 02                  ascBlockIndex1 (start with timebaseAscActionNumber =2)
-- 02                  ascBlockQuantity1 (## of actions =2)
--                  SEQUENCE OF
-- 01 02               quantity of items (ascBlockQuantity1)
--                  SEQUENCE # 1 (timebaseAscActionNumber =2)
-- 02                  timebaseAscPattern.2           (pat 2)
-- 00                  timebaseAscAuxillaryFunction.2 (bits)
-- 00                  timebaseAscSpecialFunction.2   (bits)
--                  SEQUENCE # 2 (timebaseAscActionNumber =3)
-- 03                  timebaseAscPattern.3           (pat 3)
-- 00                  timebaseAscAuxillaryFunction.3 (bits)
-- 00                  timebaseAscSpecialFunction.3   (bits)
```

3.8 PREEMPT BLOCK DATA

```
-- ascBlockData values for standard Block
-- Preempt Data shall be as follows:

AscPreemptBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x06 preempt data
    ascBlockIndex1       INTEGER (0..255),  -- preemptNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of preempts

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscPreemptBlockData
   }

AscPreemptBlockData ::= SEQUENCE
   {
    preemptControl.x           INTEGER (0..255),
    preemptLink.x              INTEGER (0..255),
    preemptDelay.x             INTEGER (0..65535),
    preemptMinimumDuration.x   INTEGER (0..65535),
    preemptMinimumGreen.x      INTEGER (0..255),
    preemptMinimumWalk.x       INTEGER (0..255),
    preemptEnterPedClear.x     INTEGER (0..255),
    preemptTrackGreen.x        INTEGER (0..255),
    preemptDwellGreen.x        INTEGER (0..255),
    preemptMaximumPresence.x   INTEGER (0..65535),
    preemptTrackPhase.x        OCTET STRING,
    preemptDwellPhase.x        OCTET STRING,
    preemptDwellPed.x          OCTET STRING,
    preemptExitPhase.x         OCTET STRING,
    preemptTrackOverlap.x      OCTET STRING,
    preemptDwellOverlap.x      OCTET STRING,
    preemptCyclingPhase.x      OCTET STRING,
    preemptCyclingPed.x        OCTET STRING,
    preemptCyclingOverlap.x    OCTET STRING,
    preemptEnterYellowChange   INTEGER (0..255),
    preemptEnterRedClear       INTEGER (0..255),
    preemptTrackYellowChange   INTEGER (0..255),
    preemptTrackRedClear       INTEGER (0..255)
   }
```

3.8.1 Preempt Block Example

```
-- The following provides an example octet string value for
-- a set or get of a preempt block.
--
--                 SEQUENCE
-- 00                 ascBlockDataType (standard block)
-- 06                 ascBlockDataID (preempt data)
-- 02                 ascBlockIndex1 (start with preemptNumber =2)
-- 02                 ascBlockQuantity1 (## of preempts=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (preemptNumber =2)
-- 05                preemptControl.2        (bits)
-- 00                preemptLink.2           (none)
--  |                etc, etc, to:
-- 28                preemptTrackYellowChange.2 (4.0 Sec)
-- 00                preemptTrackRedClear.2     (  0 Sec)
--                SEQUENCE # 2 (preemptNumber =3)
-- 05                preemptControl.3        (bits)
-- 01                preemptLink.3           (pe 1)
--  |                etc, etc, to:
-- 28                preemptTrackYellowChange.3 (4.0 Sec)
-- 00                preemptTrackRedClear.3     (  0 Sec)
```

3.9 SEQUENCE BLOCK DATA

```
-- ascBlockData values for standard Block
-- Sequence Data shall be as follows:

AscSequenceBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x07 sequence data
    ascBlockIndex1       INTEGER (0..255),  -- sequenceRingNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of rings
    ascBlockIndex2       INTEGER (0..255),  -- sequenceNumber
    ascBlockQuantity2    INTEGER (0..255),  -- ## of sequences

    -- for (
    --      y = ascBlockIndex2;
    --      y < (ascBlockIndex2 + ascBlockQuantity2);
    --      y++)
    --      for (
    --           x = ascBlockIndex1;
    --           x < (ascBlockIndex1 + ascBlockQuantity1);
    --           x++)

    data    SEQUENCE OF AscSequenceBlockData
   }

AscSequenceBlockData ::= SEQUENCE
   {
    sequenceData.y.x    OCTET STRING
   }
```

3.9.1 Sequence Block Example

```
-- The following provides an example octet string value for
-- a set or get of a sequence block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 07                ascBlockDataID (sequence data)
-- 01                ascBlockIndex1 (start with sequenceRingNumber=1)
-- 02                ascBlockQuantity1 (## of rings=2)
-- 01                ascBlockIndex2 (start with sequenceNumber=1)
-- 02                ascBlockQuantity2 (## of sequences =2)
--                  SEQUENCE OF
-- 01 04               quantity of items (ascBlockQuantity1 * ascBlockQuantity2)
--                  SEQUENCE # 1 (sequenceNumber=1 / sequenceRingNumber=1)
-- 04 01 02 03 04      sequenceData.1.1 (ph 1-2-3-4)
--                  SEQUENCE # 2 (sequenceNumber=1 / sequenceRingNumber=2)
-- 04 05 06 07 08      sequenceData.1.2 (ph 5-6-7-8)
--                  SEQUENCE # 3 (sequenceNumber=2 / sequenceRingNumber=1)
-- 04 02 01 04 03      sequenceData.2.1 (ph 1-2-3-4)
--                  SEQUENCE # 4 (sequenceNumber=2 / sequenceRingNumber=2)
-- 04 06 05 08 07      sequenceData.2.2 (ph 5-6-7-8)
```

3.10 CHANNEL BLOCK DATA

```
-- ascBlockData values for standard Block
-- Channel Data shall be as follows:

AscChannelBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x08 channel data
    ascBlockIndex1       INTEGER (0..255),  -- channelNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of channels

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscChannelBlockData
   }

AscChannelBlockData ::= SEQUENCE
   {
    channelControlSource.x   INTEGER (0..255),
    channelControlType.x     INTEGER (1..4),
    channelFlash.x           INTEGER (0..255),
    channelDim.x             INTEGER (0..255)
   }
```

3.10.1 Channel Block Example

```
-- The following provides an example octet string value for
-- a SET or GET of a channel block.
--
--                  SEQUENCE
-- 00                  ascBlockDataType (standard block)
-- 08                  ascBlockDataID (channel data)
-- 02                  ascBlockIndex1 (start with channelNumber=2)
-- 02                  ascBlockQuantity1 (## of channels=2)
--                  SEQUENCE OF
-- 01 02               quantity of items (ascBlockQuantity1)
--                  SEQUENCE # 1 (channelNumber=2)
-- 02                  channelControlSource.2 (ph 2)
-- 02                  channelControlType.2   (phaseVehicle)
-- 02                  channelFlash.2         (bits)
-- 07                  channelDim.2           (bits)
--                  SEQUENCE # 2 (channelNumber=3)
-- 03                  channelControlSource.3 (ph 3)
-- 02                  channelControlType.3   (phaseVehicle)
-- 04                  channelFlash.3         (bits)
-- 0F                  channelDim.3           (bits)
```

3.11 OVERLAP BLOCK DATA

```
-- ascBlockData values for standard Block
-- Overlap Data shall be as follows:

AscOverlapBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x09 overlap data
    ascBlockIndex1       INTEGER (0..255),  -- overlapNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of overlaps

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscOverlapBlockData
   }

AscOverlapBlockData ::= SEQUENCE
   {
    overlapIncludedPhases.x   OCTET STRING,
    overlapModifierPhases.x   OCTET STRING,
    overlapTrailGreen.x       INTEGER (0..255),
    overlapTrailYellow.x      INTEGER (0..255),
    overlapTrailRed.x         INTEGER (0..255)
   }
```

3.11.1 Overlap Block Example

```
-- The following provides an example octet string value for
-- a SET or GET of a overlap block.
--
--                  SEQUENCE
-- 00                  ascBlockDataType (standard block)
-- 09                  ascBlockDataID (overlap data)
-- 02                  ascBlockIndex1 (start with overlapNumber=2)
-- 02                  ascBlockQuantity1 (## of overlaps=2)
--                  SEQUENCE OF
-- 01 02               quantity of items (ascBlockQuantity1)
--                  SEQUENCE # 1 (overlapNumber=2)
-- 02 02 03            overlapIncludedPhases.2 (ph 2 & 3)
-- 00                  overlapModifierPhases.2 (none)
-- 00                  overlapTrailGreen.2     (0 sec)
-- 23                  overlapTrailYellow.2    (3.5 sec)
-- 05                  overlapTrailRed.2       (0.5 sec)
--                  SEQUENCE # 2 (overlapNumber=3)
-- 02 04 05            overlapIncludedPhases.3 (ph 4 & 5)
-- 00                  overlapModifierPhases.3 (none)
-- 00                  overlapTrailGreen.3     (0 sec)
-- 23                  overlapTrailYellow.3    (3.5 sec)
-- 05                  overlapTrailRed.3       (0.5 sec)
```

3.12 PORT 1 BLOCK DATA

```
-- ascBlockData values for standard Block
-- Port 1 Data shall be as follows:

AscPort1Block ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x0A port 1 data
    ascBlockIndex1       INTEGER (0..255),  -- port1Number
    ascBlockQuantity1    INTEGER (0..255),  -- ## of address

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscPort1BlockData
   }

AscPort1BlockData ::= SEQUENCE
   {
    port1DevicePresent.x   INTEGER (0..1),
    port1Frame40Enable.x   INTEGER (0..1)
   }
```

3.12.1 Port 1 Block Example

```
-- The following provides an example octet string value for
-- a SET or GET of a port 1 block.
--
--                  SEQUENCE
-- 00                  ascBlockDataType (standard block)
-- 0A                  ascBlockDataID (port 1 data)
-- 02                  ascBlockIndex1 (start with port1Number=2)
-- 02                  ascBlockQuantity1 (## of address=2)
--                  SEQUENCE OF
-- 01 02               quantity of items (ascBlockQuantity1)
--                  SEQUENCE # 1 (port1Number=2)
-- 01                  port1DevicePresent.2 (true)
-- 00                  port1Frame40Enable.2 (false)
--                  SEQUENCE # 2 (port1Number=3)
-- 01                  port1DevicePresent.3 (true)
-- 00                  port1Frame40Enable.3 (false)
```

3.13 SCHEDULE BLOCK DATA

```
-- ascBlockData values for standard Block
-- Schedule Data shall be as follows:

AscScheduleBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x0B schedule data
    ascBlockIndex1       INTEGER (0..255),  -- timeBaseScheduleNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of schedules
    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscScheduleBlockData
   }

AscScheduleBlockData ::= SEQUENCE
   {
    timeBaseScheduleMonth.x     INTEGER (0..65535),
    timeBaseScheduleDay.x       INTEGER (0..255),
    timeBaseScheduleDate.x      INTEGER (0..4294967295),
    timeBaseScheduleDayPlan.x   INTEGER (1..255)
   }
```

3.13.1 Schedule Block Example

```
-- The following provides an example octet string value for
-- a set or get of a schedule block.
--
--                  SEQUENCE
-- 00                  ascBlockDataType (standard block)
-- 0B                  ascBlockDataID (schedule data)
-- 02                  ascBlockIndex1 (start with timeBaseScheduleNumber=2)
-- 02                  ascBlockQuantity1 (## of schedules=2)
--                  SEQUENCE OF
-- 01 02               quantity of items (ascBlockQuantity1)
--                  SEQUENCE # 1 (timeBaseScheduleNumber=2)
-- 1F FE               timeBaseScheduleMonth.2   (all)
-- 04                  timeBaseScheduleDay.2     (Mon)
-- FF FF FF FE         timeBaseScheduleDate.2    (all)
-- 02                  timeBaseScheduleDayPlan.2 (dp 2)
--                  SEQUENCE # 2 (timeBaseScheduleNumber=3)
-- 1F FE               timeBaseScheduleMonth.3   (all)
-- 08                  timeBaseScheduleDay.3     (Tue)
-- FF FF FF FE         timeBaseScheduleDate.3    (all)
-- 03                  timeBaseScheduleDayPlan.3 (dp 3)
```

3.14 DAY PLAN BLOCK DATA

```
-- ascBlockData values for standard Block
-- Day Plan Data shall be as follows:

AscDayPlanBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x0C day plan data
    ascBlockIndex1       INTEGER (0..255),  -- dayPlanEventNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of day plan events
    ascBlockIndex2       INTEGER (0..255),  -- dayPlanNumber
    ascBlockQuantity2    INTEGER (0..255),  -- ## of day plans

    -- for (
    --      y = ascBlockIndex2;
    --      y < (ascBlockIndex2 + ascBlockQuantity2);
    --      y++)
    --      for (
    --           x = ascBlockIndex1;
    --           x < (ascBlockIndex1 + ascBlockQuantity1);
    --           x++)

    data    SEQUENCE OF AscDayPlanBlockData
   }

AscDayPlanBlockData ::= SEQUENCE
   {
    dayPlanHour.y.x              INTEGER (0..23),
    dayPlanMinute.y.x            INTEGER (0..59),
    dayPlanActionNumberOID.y.x   OBJECT IDENTIFIER
   }
```

3.14.1 Day Plan Block Example

```
-- The following provides an example octet string value for
-- a SET or - of a day plan block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 0C                ascBlockDataID (day plan data)
-- 01                ascBlockIndex1 (start with dayPlanEventNumber=1)
-- 02                ascBlockQuantity1 (## of day plan events=2)
-- 01                ascBlockIndex2 (start with dayPlanNumber=1)
-- 02                ascBlockQuantity2 (## of day plans=2)
--                SEQUENCE OF
-- 01 04             quantity of items (ascBlockQuantity1 * ascBlockQuantity2)
--                SEQUENCE # 1 (dayPlanNumber=1 / dayPlanEventNumber=1)
-- 04                dayPlanHour.1.1 (04 hours)
-- 30                dayPlanMinute.1.1 (30 minutes)
--                   dayPlanActionNumberOID.1.1 (timebaseAscActionNumber=1)
-- 0F 2B 06 01 04 01 89 36 04 02 01 05 03 01 01 01
--                SEQUENCE # 2 (dayPlanNumber=1 / dayPlanEventNumber=2)
-- 06                dayPlanHour.1.2 (06 hours)
-- 00                dayPlanMinute.1.2 (00 minutes)
--                   dayPlanActionNumberOID.1.2 (timebaseAscActionNumber=2)
-- 0F 2B 06 01 04 01 89 36 04 02 01 05 03 01 01 02
--                SEQUENCE # 3 (dayPlanNumber=2 / dayPlanEventNumber=1)
-- 05                dayPlanHour.2.1 (05 hours)
-- 30                dayPlanMinute.2.1 (30 minutes)
--                   dayPlanActionNumberOID.2.1 (timebaseAscActionNumber=1)
-- 0F 2B 06 01 04 01 89 36 04 02 01 05 03 01 01 01
--                SEQUENCE # 4 (dayPlanNumber=2 / dayPlanEventNumber=2)
-- 08                dayPlanHour.2.2 (08 hours)
-- 00                dayPlanMinute.2.2 (00 minutes)
--                   dayPlanActionNumberOID.2.2 (timebaseAscActionNumber=2)
-- 0F 2B 06 01 04 01 89 36 04 02 01 05 03 01 01 02
```

3.15 EVENT LOG CONFIG BLOCK DATA

```
-- ascBlockData values for standard Block
-- Event Config Data shall be as follows:

AscEventConfigBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x0D event log config data
    ascBlockIndex1       INTEGER (0..255),  -- eventConfigID
    ascBlockQuantity1    INTEGER (0..255),  -- ## of events
    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscEventConfigBlockData
   }

AscEventConfigBlockData ::= SEQUENCE
   {
    eventConfigClass.x           INTEGER (1..255),
    eventConfigMode.x            INTEGER (1..6),
    eventConfigCompareValue.x    INTEGER,
    eventConfigCompareValue2.x   INTEGER,
    eventConfigCompareOID.x      OBJECT IDENTIFIER,
    eventConfigLogOID.x          OBJECT IDENTIFIER,
    eventConfigAction.x          INTEGER (1..3)
   }
```

3.15.1 Event Log Config Block Example

```
-- The following provides an example octet string value for
-- a set or get of a event log config block.
--
--                 SEQUENCE
-- 00                 ascBlockDataType (standard block)
-- 0D                 ascBlockDataID (event log config data)
-- 02                 ascBlockIndex1 (start with eventConfigID=2)
-- 02                 ascBlockQuantity1 (## of events=2)
--                 SEQUENCE OF
-- 01 02              quantity of items (ascBlockQuantity1)
--                 SEQUENCE # 1 (eventConfigID=2)
-- 01                 eventConfigClass.2         (class=1)
-- 02                 eventConfigMode.2          (onChange)
-- 00                 eventConfigCompareValue.2  (no value)
-- 00                 eventConfigCompareValue2.2 (no value)
--                    eventConfigCompareOID.2    (shortAlarmStatus.0)
-- 0D 2B 06 01 04 01 89 36 04 02 01 03 09 00
--                    eventConfigLogOID.2        (shortAlarmStatus.0)
-- 0D 2B 06 01 04 01 89 36 04 02 01 03 09 00
-- 03                 eventConfigAction.2        (log)
--                 SEQUENCE # 2 (eventConfigID=3)
-- 01                 eventConfigClass.3         (class=1)
-- 02                 eventConfigMode.3          (onChange)
-- 00                 eventConfigCompareValue.3  (no value)
-- 00                 eventConfigCompareValue2.3 (no value)
--                    eventConfigCompareOID.3    (unitAlarmStatus1.0)
-- 0D 2B 06 01 04 01 89 36 04 02 01 03 08 00
--                    eventConfigLogOID.3        (unitAlarmStatus1.0)
-- 0D 2B 06 01 04 01 89 36 04 02 01 03 08 00
-- 03                 eventConfigAction.3        (log)
```

3.16 EVENT CLASS BLOCK DATA

```
-- ascBlockData values for standard Block
-- Event Class Data shall be as follows:

AscEventClassBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x0E event class data
    ascBlockIndex1       INTEGER (0..255),  -- eventClassNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of classes
    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscEventClassBlockData
   }

AscEventClassBlockData ::= SEQUENCE
   {
    eventClassLimit.x          INTEGER (0..255),
    eventClassClearTime.x      Counter,
    eventClassDescription.x    OCTET STRING
   }
```

3.16.1 Event Class Block Example

```
-- The following provides an example octet string value for
-- a set or get of a event class block.

-- Note – the sum of all eventClassLimit values can not be
-- greater than maxEventLogSize. The values may need to be
-- set to zero prior to setting new values.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 0E                ascBlockDataID (event class data)
-- 02                ascBlockIndex1 (start with eventClassNumber=2)
-- 02                ascBlockQuantity1 (## of classes=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (eventClassNumber=2)
-- 0A                eventClassLimit.2 (10)
-- 00 00 00 00       eventClassClearTime.2 (00:00:00 01/01/1970)
--                   eventClassDescription.2 (Class 2)
-- 07 43 6C 61 73 73 20 32
--                SEQUENCE # 2 (eventClassNumber=3)
-- 0A                eventClassLimit.3 (10)
-- 00 00 00 00       eventClassClearTime.3 (00:00:00 01/01/1970)
--                   eventClassDescription.3 (Class 3)
-- 07 43 6C 61 73 73 20 33
```

3.17 DYNAMIC OBJECT CONFIG BLOCK DATA

```
-- ascBlockData values for standard Block
-- Dynamic Object Config Data shall be as follows:

AscDynObjConfigBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x0F dyn obj config data
    ascBlockIndex1       INTEGER (0..255),  -- dynObjIndex
    ascBlockQuantity1    INTEGER (0..255),  -- ## of indexes
    ascBlockIndex2       INTEGER (0..255),  -- dynObjNumber
    ascBlockQuantity2    INTEGER (0..255),  -- ## of dyn objects
    -- for (
    --      y = ascBlockIndex2;
    --      y < (ascBlockIndex2 + ascBlockQuantity2);
    --      y++)
    --      for (
    --           x = ascBlockIndex1;
    --           x < (ascBlockIndex1 + ascBlockQuantity1);
    --           x++)

    data    SEQUENCE OF AscDynObjConfigBlockData
   }

AscDynObjConfigBlockData ::= SEQUENCE
   {
    dynObjVariable.y.x   OBJECT IDENTIFIER
   }
```

3.17.1 Dynamic Object Config Block Example

```
-- The following provides an example octet string value for
-- a set or get of a dynamic object config block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 0F                ascBlockDataID (dyn obj config data)
-- 01                ascBlockIndex1 (start with dynObjIndex=1)
-- 02                ascBlockQuantity1 (## of indexes=2)
-- 01                ascBlockIndex2 (start with dynObjNumber=1)
-- 02                ascBlockQuantity2 (## of dyn objects=2)
--                SEQUENCE OF
-- 01 04             quantity of items (ascBlockQuantity1 * ascBlockQuantity2)
--                SEQUENCE # 1 (dynObjNumber=1 / dynObjIndex=1)
--                   dynObjVariable.1.1 (coordPatternStatus.0)
-- 0D 2B 06 01 04 01 89 36 04 02 01 04 0A 00
--                SEQUENCE # 2 (dynObjNumber=1 / dynObjIndex=2)
--                   dynObjVariable.1.2 (coordCycleStatus.0)
-- 0D 2B 06 01 04 01 89 36 04 02 01 04 0C 00
--                SEQUENCE # 3 (dynObjNumber=2 / dynObjIndex=1)
--                   dynObjVariable.2.1 (volumeOccupancySequence.0)
-- 0E 2B 06 01 04 01 89 36 04 02 01 02 05 01 00
--                SEQUENCE # 4 (dynObjNumber=2 / dynObjIndex=2)
--                   dynObjVariable.2.2 (volumeOccupancyPeriod.0)
-- 0E 2B 06 01 04 01 89 36 04 02 01 02 05 02 00
```

3.18 DYNAMIC OBJECT OWNER BLOCK DATA

```
-- ascBlockData values for standard Block
-- Dynamic Object Owner Data shall be as follows:

AscDynObjOwnerBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x10 dyn obj owner data
    ascBlockIndex1       INTEGER (0..255),  -- dynObjNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of dyn obj
    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscDynObjOwnerBlockData
   }

AscDynObjOwnerBlockData ::= SEQUENCE
   {
    dynObjConfigOwner.x    OwnerString
   }
```

3.18.1 Dynamic Object Owner Block Example

```
-- The following provides an example octet string value for
-- a set or get of a dynamic object owner block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 10                ascBlockDataID (dyn obj owner data)
-- 02                ascBlockIndex1 (start with dynObjNumber=2)
-- 02                ascBlockQuantity1 (## of dyn obj=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (dynObjNumber=2)
--                   dynObjConfigOwner.2 (TMC 2)
-- 05 54 4D 43 20 32
--                SEQUENCE # 2 (dynObjNumber=3)
--                   dynObjConfigOwner.3 (TMC 2)
-- 05 54 4D 43 20 32
```

3.19 DYNAMIC OBJECT STATUS BLOCK DATA

```
-- ascBlockData values for standard Block
-- Dynamic Object Status Data shall be as follows:

AscDynObjStatusBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x11 dyn obj status data
    ascBlockIndex1       INTEGER (0..255),  -- dynObjNumber
    ascBlockQuantity1    INTEGER (0..255),  -- ## of dyn obj

    -- for (
    --      x = ascBlockIndex1;
    --      x < (ascBlockIndex1 + ascBlockQuantity1);
    --      x++)

    data    SEQUENCE OF AscDynObjStatusBlockData
   }

AscDynObjStatusBlockData ::= SEQUENCE
   {
    dynObjConfigStatus.x    ConfigEntryStatus
   }
```

3.19.1 Dynamic Object Status Block Example

```
-- The following provides an example octet string value for
-- a set or get of a dynamic object status block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 11                ascBlockDataID (dyn obj status data)
-- 02                ascBlockIndex1 (start with dynObjNumber=2)
-- 02                ascBlockQuantity1 (## of dyn obj=2)
--                SEQUENCE OF
-- 01 02             quantity of items (ascBlockQuantity1)
--                SEQUENCE # 1 (dynObjNumber=2)
-- 01                dynObjConfigStatus.2 (valid)
--                SEQUENCE # 2 (dynObjNumber=3)
-- 01                dynObjConfigStatus.3 (valid)
```

3.20 MISCELLANEOUS ASC BLOCK DATA

```
-- ascBlockData values for standard Block
-- Misc ASC Data shall be as follows:

AscMiscBlock ::= SEQUENCE
   {
    ascBlockDataType     INTEGER (0..255),  -- 0x00 standard block
    ascBlockDataID       INTEGER (0..255),  -- 0x12 misc ASC data

    data    SEQUENCE OF AscMiscBlockData
   }

AscMiscBlockData ::= SEQUENCE
   {
    dynamicObjectPersistence.0    INTEGER (0..65535),
    volumeOccupancyPeriod.0       INTEGER (0..255),
    unitStartUpFlash.0            INTEGER (0..255),
    unitAutoPedestrianClear.0     INTEGER (1..2),
    unitBackupTime.0              INTEGER (0..65535),
    unitRedRevert.0               INTEGER (0..255),
    coordOperationalMode.0        INTEGER (0..255),
    coordCorrectionMode.0         INTEGER (1..4),
    coordMaximumMode.0            INTEGER (1..4),
    coordForceMode.0              INTEGER (1..3),
    timebaseAscPatternSync.0      INTEGER (0..65535),
    globalDayLightSavings.0       INTEGER (1..3),
    controller-standardTimeZone.0 INTEGER (-43200..43200)
   }
```

3.20.1 Miscellaneous ASC Block Example

```
-- The following provides an example octet string value for
-- a set or get of a miscellaneous asc block.
--
--                SEQUENCE
-- 00                ascBlockDataType (standard block)
-- 12                ascBlockDataID (misc asc data)
--                SEQUENCE OF
-- 01 01             quantity of items
--                SEQUENCE # 1
-- 00 F0             dynamicObjectPersistence.0    (240 sec)
-- 1E                volumeOccupancyPeriod.0       (30 sec)
-- 05                unitStartUpFlash.0            (5 sec)
-- 02                unitAutoPedestrianClear.0     (enable)
-- 03 84             unitBackupTime.0              (900 sec)
-- 14                unitRedRevert.0               (20 tSec)
-- 00                coordOperationalMode.0        (auto)
-- 03                coordCorrectionMode.0         (sw)
-- 04                coordMaximumMode.0            (inh)
-- 02                coordForceMode.0              (float)
-- 00 00             timebaseAscPatternSync.0      (midnight)
-- 03                globalDayLightSavings.0       (enableUS)
-- FF FF B9 B0       controller-standardTimeZone.0 (-18000 sec)
```

Annex A
INFORMATION PROFILE
(Normative)

A conformance group is a basic unit of conformance and is used to specify a collection of related managed objects. The conformance group designation applied to a set of objects provides a systematic means for determining which objects are required to support a function. If a device has multiple functions, a Conformance Group will be defined for each function. Conformance group definitions will be found in the NTCIP Object Definition Standard documents. The Object Definition Standard may define a Conformance Group with objects that are not in lexicographic order and only apply to devices of that type.

The related managed objects of a conformance group may include mandatory and/or optional objects. Mandatory objects within a conformance group shall be implemented. Optional objects shall be implemented only if a defined function of the device requires that particular object.

For example, assume a device implements an asynchronous RS-232 interface. It must implement all the mandatory objects in the asynchronous conformance group of the RS-232 MIB. It would not have to implement the Synchronous Conformance Group of objects unless it also provided a synchronous interface.

Assume also that the Asynchronous Conformance Group has a *CRC error counter* object that is optional. The *CRC error counter* object would not have to be implemented unless the device used CRC checking on the asynchronous interface.

Conformance groups are defined as either mandatory or optional. If a conformance group is mandatory, all of the objects with STATUS "mandatory" that are part of the Conformance Group shall be present for a device to claim conformance to the Conformance Group. If a Conformance Group is optional, all of the objects that are part of the Conformance Group with the STATUS "mandatory" shall be present if the device supports the Conformance Group. Objects with the STATUS "optional" may be supported.

When a table is included in a conformance group, all objects contained in the table are included by reference. This is because a table is defined as a SEQUENCE OF {SEQUENCE}. Thus, all objects listed in the sequence are defined as an integral part of the table. Tables are defined as either mandatory or optional. If a table is mandatory, all of the objects with STATUS "mandatory" shall be present. If a table is optional, all of the objects with the STATUS "mandatory" shall be present if the device supports the table. Objects in the table with the STATUS "optional" may be supported.

A.1 NOTATION

The following notations and symbols are used to indicate status and conditional status within this standard.

A.1.1 Type Symbols

The following symbols are used to indicate type:

Symbol	Type
C	Control Object - use of 'dbCreateTransaction' in NTCIP 1201 Clause 2.3.1 shall NOT delay a SET to this object.
P	Parameter Object - use of 'dbCreateTransaction' in NTCIP 1201 Clause 2.3.1 to SET this object is optional. NOTE—The device must support both the normal SNMP SET and a SET via dbCreateTransaction.
P2	Parameter Object - use of 'dbCreateTransaction' in NTCIP 1201 Clause 2.3.1 to SET this object is mandatory. NOTE—The device must NOT allow a normal SNMP SET.
S	Status / Information Object - this object is read only therefore a SET is not permitted.

A.1.2 Status Symbols

The following symbols are used to indicate status:

Symbol	Status
M	Mandatory
M.<n>	Support of every item of the group labeled by the same numeral <n> required, but only one is active at time.
O	Optional
O.<n>	Optional, but support of at least one of the group of options labeled by the same numeral <n> is required
C	Conditional
N/A	Non-applicable (i.e., logically impossible in the scope of the profile)
X	Excluded or prohibited

A.1.3 Conditional Status Notation

The following predicate notations is used:

Notation	Status
"<predicate>: M	Item is conditional on the <predicate>.

The <predicate>: notation means that the Status following it applies only when the feature or features identified by the predicate are supported. In the simplest case, <predicate> is the identifying tag of a single item.

A.1.4 Support Column

This section is in the form of a PICS and, therefore, includes a support column. An implementer claims support of an item by circling the appropriate answer (Yes or No) in the support column:

A.2 ASC REQUIREMENTS

The Conformance Group definitions for Actuated Signal Controllers are defined in this clause. An Actuated Signal Controller has multiple functions; thus, Conformance Groups are defined for each function.

The following table lists functional requirements for an Actuated Signal Controller, and asks if the listed features have been implemented.

Ref	Areas	Clause of Profile	Status	Support
A.3	Phase Conformance Group	NTCIP 1202 - 2.2	M	Yes
A.4	Detector Conformance Group	NTCIP 1202 - 2.3	M	Yes
A.5	Volume Occupancy Report Conformance Group	NTCIP 1202 - 2.3	O	Yes / No
A.6	Unit Conformance Group	NTCIP 1202 - 2.4	O	Yes / No
A.7	Special Function Conformance Group	NTCIP 1202 - 2.4	O	Yes / No
A.8	Coordination Conformance Group	NTCIP 1202 - 2.5	O	Yes / No
A.9	Time Base Conformance Group	NTCIP 1202 - 2.6	O	Yes / No
A.10	Preempt Conformance Group	NTCIP 1202 - 2.7	O	Yes / No
A.11	Ring Conformance Group	NTCIP 1202 - 2.8	O	Yes / No
A.12	Channel Conformance Group	NTCIP 1202 - 2.9	O	Yes / No
A.13	Overlap Conformance Group	NTCIP 1202 - 2.10	O	Yes / No
A.14	TS 2 Port 1 Conformance Group	NTCIP 1202 - 2.11	O	Yes / No
A.15	Block Object Conformance Group	NTCIP 1202 - 2.12	O	Yes / No
A.16	Configuration Conformance Group	NTCIP 1201 - 2.2	M	Yes
A.17	Database Management Conformance Group	NTCIP 1201 - 2.3	M	Yes
A.18	Report Conformance Group	NTCIP 1201 – 2.5	O	Yes / No
A.19	Auxiliary I/O Group	NTCIP 1201 – 2.8	N/A	No
A.20	PMPP Group	NTCIP1201 - 2.6	O	Yes / No
A.21	SNMP Group	rfc1213	M	Yes
A.22	System Group	rfc1213	M	Yes
A.23	SFMP Group	NTCIP 1103 - A.4	NA	No
A.24	STMP Group	NTCIP 1103 - A.5	O	Yes / No
A.25	Logical Name Group	NTCIP 1103 - A.6	O	Yes / No
A.26	Trap Management Group	NTCIP 1103 - A.7-9	O	Yes / No
A.27	Security Group	NTCIP 1103 - A.10	M	Yes
A.28	RS232 Group	rfc1317	O	Yes / No
A.29	HDLC Group	rfc1381	O	Yes / No
A.30	Interfaces Group	rfc1213	O	Yes / No
A.31	IP Group	rfc1213	O	Yes / No
A.32	ICMP Group	rfc1213	O	Yes / No
A.33	TCP Group	rfc1213	O	Yes / No
A.34	UDP Group	rfc1213	O	Yes / No
A.35	Ethernet Group	rfc1643	O	Yes / No

Actuated Signal Controller (ASC) devices shall adhere to the conformance requirements specified in the above as a minimum to claim compliance to this standard. If a device supports the functionality defined within a group, then the device shall implement the functionality in the standard format. Additional objects or groups may be supported without being non-compliant with ASC objects or NTCIP. Minimum and maximum ranges of objects that differ from the values of the object's SYNTAX field may be enforced by an application running on a device.

A device which enforces range limits within the bounds specified by the values of the object's SYNTAX field shall not be categorized as being non-compliant with ASC objects or NTCIP.

A device which supports a subset of objects with enumerated values shall not be categorized as being non-compliant with ASC objects or NTCIP.

A.3 PHASE CONFORMANCE GROUP

The Phase Conformance Group shall consist of the following objects:

PHASE CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.2	**Phase Conformance Group**	---	M	Yes	---	---
2.2.1	maxPhases	S	M	Yes	2-255	
2.2.2	phaseTable	--	M	Yes	----	---
	phaseEntry	--	M	Yes	---	---
2.2.2.1	phaseNumber	S	M	Yes	1..255	
2.2.2.2	phaseWalk	P	M	Yes	0-255	
2.2.2.3	phasePedestrianClear	P	M	Yes	0-255	
2.2.2.4	phaseMinimumGreen	P	M	Yes	0-255	
2.2.2.5	phasePassage	P	M	Yes	0-255	
2.2.2.6	phaseMaximum1	P	M	Yes	0-255	
2.2.2.7	phaseMaximum2	P	M	Yes	0-255	
2.2.2.8	phaseYellowChange	P	M	Yes	0-255	
2.2.2.9	phaseRedClear	P	M	Yes	0-255	
2.2.2.10	phaseRedRevert	P	O	Yes / No	0-255	
2.2.2.11	phaseAddedInitial	P	M	Yes	0-255	
2.2.2.12	phaseMaximumInitial	P	M	Yes	0-255	
2.2.2.13	phaseTimeBeforeReduction	P	M	Yes	0-255	
2.2.2.14	phaseCarsBeforeReduction	P	O	Yes / No	0-255	
2.2.2.15	phaseTimeToReduce	P	M	Yes	0-255	
2.2.2.16	phaseReduceBy	P	O	Yes / No	0-255	
2.2.2.17	phaseMinimumGap	P	M	Yes	0-255	
2.2.2.18	phaseDynamicMaxLimit	P	O	Yes / No	0-255	
2.2.2.19	phaseDynamicMaxStep	P	O	Yes / No	0-255	
2.2.2.20	phaseStartup	P2	M	Yes	1-6	
	other(1)	--	---	Yes / No	---	---
	phaseNotON(2)	--	---	Yes / No	---	---
	greenWalk(3)	--	---	Yes / No	---	---
	greenNoWalk(4)	--	---	Yes / No	---	---
	yellowChange(5)	--	---	Yes / No	---	---
	redClear(6)	--	---	Yes / No	---	---
2.2.2.21	phaseOptions	P2	M	Yes	0-65535	
	Bit 0 - Enabled Phase	--	---	Yes / No	---	---
	Bit 1 - Automatic Flash Entry Phase	--	---	Yes / No	---	---
	Bit 2 - Automatic Flash Exit Phase	--	---	Yes / No	---	---
	Bit 3 - Non-Actuated 1	--	---	Yes / No	---	---
	Bit 4 - Non-Actuated 2	--	---	Yes / No	---	---
	Bit 5 - Non-Locking Detector Memory	--	---	Yes / No	---	---
	Bit 6 - Min Vehicle Recall	--	---	Yes / No	---	---
	Bit 7 - Max Vehicle Recall	--	---	Yes / No	---	---
	Bit 8 - Ped Recall	--	---	Yes / No	---	---
	Bit 9 - Soft Vehicle Recall	--	---	Yes / No	---	---
	Bit 10 - Dual Entry Phase	--	---	Yes / No	---	---
	Bit 11 - Simultaneous Gap Disable	--	---	Yes / No	---	---
	Bit 12 - Guaranteed Passage	--	---	Yes / No	---	---
	Bit 13 - Actuated Rest In Walk	--	---	Yes / No	---	---
	Bit 14 - Conditional Service Enable	--	---	Yes / No	---	---
	Bit 15 - Added Initial Calculation	--	---	Yes / No	---	---
2.2.2.22	phaseRing	P2	M	Yes	0-255	
2.2.2.23	phaseConcurrency	P2	M	Yes	string	
2.2.3	maxPhaseGroups	S	M	Yes	1-255	
2.2.4	phaseStatusGroupTable	--	M	Yes	----	---
	phaseStatusGroupEntry	--	M	Yes	---	---
2.2.4.1	phaseStatusGroupNumber	S	M	Yes	1-255	
2.2.4.2	phaseStatusGroupReds	S	M	Yes	0-255	
2.2.4.3	phaseStatusGroupYellows	S	M	Yes	0-255	
2.2.4.4	phaseStatusGroupGreens	S	M	Yes	0-255	

PHASE CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.2.4.5	phaseStatusGroupDontWalks	S	M	Yes	0-255	
2.2.4.6	phaseStatusGroupPedClears	S	M	Yes	0-255	
2.2.4.7	phaseStatusGroupWalks	S	M	Yes	0-255	
2.2.4.8	phaseStatusGroupVehCalls	S	M	Yes	0-255	
2.2.4.9	phaseStatusGroupPedCalls	S	M	Yes	0-255	
2.2.4.10	phaseStatusGroupPhaseOns	S	M	Yes	0-255	
2.2.4.11	phaseStatusGroupPhaseNexts	S	M	Yes	0-255	
2.2.5	phaseControlGroupTable	--	O	Yes / No	----	---
	phaseControlGroupEntry	--	2.2.5 : M	Yes	---	---
2.2.5.1	phaseControlGroupNumber	S	2.2.5 : M	Yes	1-255	
2.2.5.2	phaseControlGroupPhaseOmit	C	2.2.5 : M	Yes	0-255	
2.2.5.3	phaseControlGroupPedOmit	C	2.2.5 : M	Yes	0-255	
2.2.5.4	phaseControlGroupHold	C	2.2.5 : M	Yes	0-255	
2.2.5.5	phaseControlGroupForceOff	C	2.2.5 : O	Yes / No	0-255	
2.2.5.6	phaseControlGroupVehCall	C	2.2.5 : M	Yes	0-255	
2.2.5.7	phaseControlGroupPedCall	C	2.2.5 : M	Yes	0-255	

A.4 DETECTOR CONFORMANCE GROUP

The Detector Conformance Group consists of the following objects:

DETECTOR CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.3	**Detector Conformance Group**	--	M	Yes	----	-----
2.3.1	maxVehicleDetectors	S	M	Yes	1-255	
2.3.2	vehicleDetectorTable	--	M	Yes	----	---
	vehicleDetectorEntry	--	M	Yes	---	---
2.3.2.1	vehicleDetectorNumber	S	M	Yes	1..255	
2.3.2.2	vehicleDetectorOptions	P	M	Yes	0-255	
	Bit 0 - Volume Detector	--	---	Yes / No	---	---
	Bit 1 - Occupancy Detector	--	---	Yes / No	---	---
	Bit 2 -Yellow Lock Call	--	---	Yes / No	---	---
	Bit 3 -Red Lock Call	--	---	Yes / No	---	---
	Bit 4 -Passage	--	---	Yes / No	---	---
	Bit 5 -Added Initial	--	---	Yes / No	---	---
	Bit 6 -Queue	--	---	Yes / No	---	---
	Bit 7 - Call	--	---	Yes / No	---	---
2.3.2.3	vehicleDetectorCallPhase	P	M	Yes	0-255	
2.3.2.4	vehicleDetectorSwitchPhase	P	M	Yes	0-255	
2.3.2.5	vehicleDetectorDelay	P	M	Yes	0-65535	
2.3.2.6	vehicleDetectorExtend	P	M	Yes	0-255	
2.3.2.7	vehicleDetectorQueueLimit	P	O	Yes / No	0-255	
2.3.2.8	vehicleDetectorNoActivity	P	M	Yes	0-255	
2.3.2.9	vehicleDetectorMaxPresence	P	M	Yes	0-255	
2.3.2.10	vehicleDetectorErraticCounts	P	M	Yes	0-255	
2.3.2.11	vehicleDetectorFailTime	P	O	Yes / No	0-255	
2.3.2.12	vehicleDetectorAlarms	S	M	Yes	0-255	
2.3.2.13	vehicleDetectorReportedAlarms	S	O	Yes / No	0-255	
2.3.2.14	vehicleDetectorReset	C	M	Yes	0-1	
2.2.3	maxVehicleDetectorStatusGroups	S	M	Yes	1-255	
2.3.4	vehicleDetectorStatusGroupTable	--	M	Yes	----	---
	vehicleDetectorStatusGroupEntry	--	M	Yes	---	---
2.3.4.1	vehicleDetectorStatusGroupNumber	S	M	Yes	1-255	
2.3.4.2	vehicleDetectorStatusGroupActive	S	M	Yes	0-255	
2.3.4.3	vehicleDetectorStatusGroupAlarms	S	M	Yes	0-255	
2.3.6	maxPedestrianDetectors	S	M	Yes	1-255	
2.3.7	pedestrianDetectorTable	--	M	Yes	----	---
	pedestrianDetectorEntry	--	M	Yes	---	---
2.3.7.1	pedestrianDetectorNumber	S	M	Yes	1..255	

DETECTOR CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.3.7.2	pedestrianDetectorCallPhase	P	M	Yes	0-255	
2.3.7.3	pedestrianDetectorNoActivity	P	M	Yes	0-255	
2.3.7.4	pedestrianDetectorMaxPresence	P	M	Yes	0-255	
2.3.7.5	pedestrianDetectorErraticCounts	P	M	Yes	0-255	
2.3.7.6	pedestrianDetectorAlarms	S	M	Yes	0-255	

A.5 VOLUME OCCUPANCY REPORT CONFORMANCE GROUP

The Volume Occupancy Report Conformance Group consists of the following objects:

VOLUME OCCUPANCY REPORT CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.3.5	**VOL / OCC Report Conformance Group**	--	O	Yes / No	----	
2.3.5.1	volumeOccupancySequence	S	2.3.5 : M	Yes	0-255	
2.3.5.2	volumeOccupancyPeriod	P	2.3.5 : M	Yes	0-255	
2.3.5.3	activeVolumeOccupancyDetectors	S	2.3.5 : M	Yes	0-255	
2.3.5.4	volumeOccupancyTable	--	2.3.5 : M	Yes	----	---
	volumeOccupancyEntry	--	2.3.5 : M	Yes	---	---
2.3.5.4.1	detectorVolume	S	2.3.5 : M	Yes	0-255	
2.3.5.4.2	detectorOccupancy	S	2.3.5 : M	Yes	0-255	

A.6 UNIT CONFORMANCE GROUP

The Unit Conformance Group shall consist of the following objects:

UNIT CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.4	**Unit Conformance Group**	--	O	Yes / No	-----	----
2.4.1	unitStartUpFlash	P	2.4 : M	Yes	0-255	
2.4.2	unitAutoPedestrianClear	P	2.4 : M	Yes	1-2	
	disable(1)	--	---	Yes / No	---	---
	enable(2)	--	---	Yes / No	---	---
2.4.3	unitBackupTime	P	2.4 : M	Yes	0-65535	
2.4.4	unitRedRevert	P	2.4 : M	Yes	0-255	
2.4.5	unitControlStatus	S	2.4 : M	Yes	1-8	
2.4.6	unitFlashStatus	S	2.4 : M	Yes	1-8	
2.4.7	unitAlarmStatus2	S	2.4 : M	Yes	0-255	
2.4.8	unitAlarmStatus1	S	2.4 : M	Yes	0-255	
2.4.9	shortAlarmStatus	S	2.4 : M	Yes	0-255	
2.4.10	unitControl	C	2.4 : M	Yes	0-255	
	Bit 0 - Reserved	--	---	---	---	---
	Bit 1 - Reserved	--	---	---	---	---
	Bit 2 - External Minimum Recall	--	---	Yes / No	---	---
	Bit 3 - Call to Non-Actuated 1	--	---	Yes / No	---	---
	Bit 4 - Call to Non-Actuated 2	--	---	Yes / No	---	---
	Bit 5 - Walk Rest Modifier	--	---	Yes / No	---	---
	Bit 6 - Interconnect	--	---	Yes / No	---	---
	Bit 7 - Dimming Enable	--	---	Yes / No	---	---
2.4.11	maxAlarmGroups	S	2.4 : M	Yes	1-255	
2.4.12	alarmGroupTable	--	2.4 : M	Yes	----	----
	alarmGroupEntry	--	2.4 : M	Yes	----	----
2.4.12.1	alarmGroupNumber	S	2.4 : M	Yes	1-255	
2.4.12.2	alarmGroupState	S	2.4 : M	Yes	0-255	

A.7 SPECIAL FUNCTION CONFORMANCE GROUP

The Special Function Conformance Group shall consist of the following objects:

SPECIAL FUNCTION CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
sfg	**Special Function Conformance Group**	--	O	Yes / No	----	---
2.4.13	maxSpecialFunctionOutputs	S	sfg : M	Yes	1-255	
2.4.14	specialFunctionOutputTable	--	sfg : M	Yes	---	---
	specialFunctionOutputEntry	--	sfg : M	Yes	---	---
2.4.14.1	specialFunctionOutputNumber	S	sfg : M	Yes	1-255	
2.4.14.3	specialFunctionOutputControl	C	sfg : M	Yes	0-1	
2.4.14.4	specialFunctionOutputStatus	S	sfg : M	Yes	0-1	

A.8 COORDINATION CONFORMANCE GROUP

The Coordination Conformance Group shall consist of the following objects:

COORDINATION CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.5	**Coordination Conformance Group**	--	O	Yes / No	-----	----
2.5.1	coordOperationalMode	P	2.5 : M	Yes	0-255	
2.5.2	coordCorrectionMode	P	2.5 : M	Yes	1-4	
	other(1)	--	---	Yes / No	---	---
	dwell(2)	--	---	Yes / No	---	---
	shortway(3)	--	---	Yes / No	---	---
	addOnly(4)	--	---	Yes / No	---	---
2.5.3	coordMaximumMode	P	2.5 : M	Yes	1-4	
	other(1)	--	---	Yes / No	---	---
	maximum1(2)	--	---	Yes / No	---	---
	maximum2(3)	--	---	Yes / No	---	---
	maxinhibit(4)	--	---	Yes / No	---	---
2.5.4	coordForceMode	P	2.5 : M	Yes	1-3	
	other(1)	--	---	Yes / No	---	---
	floating(2)	--	---	Yes / No	---	---
	fixed(3)	--	---	Yes / No	---	---
2.5.5	maxPatterns	S	2.5 : M	Yes	1-253	
2.5.6	patternTableType	S	2.5 : M	Yes	1-4	
	other(1)	--	---	Yes / No	---	---
	patterns(2)	--	---	Yes / No	---	---
	offset3(3)	--	---	Yes / No	---	---
	offset5(4)	--	---	Yes / No	---	---
2.5.7	patternTable	--	2.5 : M	Yes	---	---
	patternEntry	--	2.5 : M	Yes	---	---
2.5.7.1	patternNumber	S	2.5 : M	Yes	1-253	
2.5.7.2	patternCycleTime	P	2.5 : M	Yes	0-255	
2.5.7.3	patternOffsetTime	P	2.5 : M	Yes	0-255	
2.5.7.4	patternSplitNumber	S	2.5 : M	Yes	1-255	
2.5.7.5	patternSequenceNumber	P	2.5 : M	Yes	1-255	
2.5.8	maxSplits	S	2.5 : M	Yes	1-255	
2.5.9	splitTable	--	2.5 : M	Yes	---	---
	splitEntry	--	2.5 : M	Yes	---	---
2.5.9.1	splitNumber	S	2.5 : M	Yes	1-255	
2.5.9.2	splitPhase	S	2.5 : M	Yes	1-255	
2.5.9.3	splitTime	P	2.5 : M	Yes	0-255	
2.5.9.4	splitMode	P	2.5 : M	Yes	1-7	
	other(1)	--	---	Yes / No	---	---
	none(2)	--	---	Yes / No	---	---
	minimumVehicleRecall(3)	--	---	Yes / No	---	---
	maximumVehicleRecall(4)	--	---	Yes / No	---	---
	pedestrianRecall(5)	--	---	Yes / No	---	---
	maximumVehicleAndPedestrainRecall(6)	--	---	Yes / No	---	---

COORDINATION CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
	phaseOmitted(7)	--	---	Yes / No	---	---
2.5.9.5	splitCoordPhase	P	2.5 : M	Yes	0-1	
2.5.10	coordPatternStatus	S	2.5 : M	Yes	0-255	
2.5.11	localFreeStatus	S	2.5 : M	Yes	1-11	
2.5.12	coordCycleStatus	S	2.5 : M	Yes	0-510	
2.5.13	coordSyncStatus	S	2.5 : M	Yes	0-510	
2.5.14	systemPatternControl	C	2.5 : M	Yes	0-255	
2.5.15	systemSyncControl	C	2.5 : M	Yes	0-255	

A.9 TIME BASE CONFORMANCE GROUP

The Time Base Conformance Group shall consist of the following objects:

TIME BASE CONFORMANCE GROUP						
NTCIP 1201 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.4	**Time Base Conformance Group**	--	O	Yes / No	-----	----
2.4.1	globalTme	C	2.4 : M	Yes	counter	
2.4.2	globalDayLightSavings	P	2.4 : M	Yes	1-19	
2.4.3	timebase	--	2.4 : M	Yes	----	---
2.4.3.1	maxTimeBaseScheduleEntries	S	2.4 : M	Yes	1-65535	
2.4.3.2	timeBaseScheduleTable	--	2.4 : M	Yes	---	---
	timeBaseScheduleEntry	--	2.4 : M	Yes	---	---
2.4.3.2.1	timeBaseScheduleNumber	S	2.4 : M	Yes	1-65535	
2.4.3.2.2	timeBaseScheduleMonth	P	2.4 : M	Yes	0-65535	
2.4.3.2.3	timeBaseScheduleDay	P	2.4 : M	Yes	0-255	
2.4.3.2.4	timeBaseScheduleDate	P	2.4 : M	Yes	0-4294967295	
2.4.3.2.5	timeBaseScheduleDayPlan	P	2.4 : M	Yes	0-255	
2.4.3.3	timeBaseScheduleTable-status	S	2.4 : M	Yes	0-65535	
2.4.4.1	maxDayPlans	S	2.4 : M	Yes	1-255	
2.4.4.2	maxDayPlanEvents	S	2.4 : M	Yes	1-255	
2.4.4.3	timeBaseDayPlanTable	--	2.4 : M	Yes	---	---
	timeBaseDayPlanEntry	--	2.4 : M	Yes	---	---
2.4.4.3.1	dayPlanNumber	S	2.4 : M	Yes	1-255	
2.4.4.3.2	dayPlanEventNumber	S	2.4 : M	Yes	1-255	
2.4.4.3.3	dayPlanHour	P	2.4 : M	Yes	0-23	
2.4.4.3.4	dayPlanMinute	P	2.4 : M	Yes	0-59	
2.4.4.3.5	dayPlanActionNumberOID	P	2.4 : M	Yes	OID	
2.4.4.4	dayPlanStatus	S	2.4 : M	Yes	0-255	
2.4.6	controllerStandardTimeZone	P	2.4 : M	Yes	-43200/43200	
2.4.7	controllerLocalTime	S	2.4 : M	Yes	counter	

TIME BASE CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.6	timebaseAsc	--	2.4 : M	Yes	----	---
2.6.1	timebaseAscPatternSync	P	2.4 : M	Yes	0-65535	
2.6.2	maxTimebaseAscActions	S	2.4 : M	Yes	1-255	
2.6.3	timebaseAscActionTable	--	2.4 : M	Yes	---	---
	timebaseAscActionEntry	--	2.4 : M	Yes	---	---
2.6.3.1	timebaseAscActionNumber	S	2.4 : M	Yes	1-255	
2.6.3.2	timebaseAscPattern	P	2.4 : M	Yes	0-255	
2.6.3.3	timebaseAscAuxillaryFunction	P	2.4 : M	Yes	0-255	
	Bit 0 - Auxiliary 1	--	---	Yes / No	---	---
	Bit 1 - Auxiliary 2	--	---	Yes / No	---	---
	Bit 2 - Auxiliary 3	--	---	Yes / No	---	---
	Bit 3 - Dimming	--	---	Yes / No	---	---
	Bit 4 - Reserved	--	---	---	---	---

TIME BASE CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
	Bit 5 - Reserved	--	---	---	---	---
	Bit 6 - Reserved	--	---	---	---	---
	Bit 7 - Reserved	--	---	---	---	---
2.6.3.4	timebaseAscSpecialFunction	P	2.4 : M	Yes	0-255	
	Bit 0 - Special Function 1	--	---	Yes / No	---	---
	Bit 1 - Special Function 2	--	---	Yes / No	---	---
	Bit 2 - Special Function 3	--	---	Yes / No	---	---
	Bit 3 - Special Function 4	--	---	Yes / No	---	---
	Bit 4 - Special Function 5	--	---	Yes / No	---	---
	Bit 5 - Special Function 6	--	---	Yes / No	---	---
	Bit 6 - Special Function 7	--	---	Yes / No	---	---
	Bit 7 - Special Function 8	--	---	Yes / No	---	---
2.6.4	timebaseAscActionStatus	S	2.4 : M	Yes	0-255	

A.10 PREEMPT CONFORMANCE GROUP

The Preempt Conformance Group shall consist of the following objects:

PREEMPT CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.7	**Preempt Conformance Group**	--	O	Yes / No	-----	----
2.7.1	maxPreempts	S	2.7 : M	Yes	1-255	
2.7.2	preemptTable	--	2.7 : M	Yes	---	---
	preemptEntry	--	2.7 : M	Yes	---	---
2.7.2.1	preemptNumber	S	2.7 : M	Yes	1-255	
2.7.2.2	preemptControl	P	2.7 : M	Yes	0-255	
	Bit 0 - Non-Locking memory	--	---	Yes / No	---	---
	Bit 1 - Override Flash	--	---	Yes / No	---	---
	Bit 2 - Override preemptNumber+1	--	---	Yes / No	---	---
	Bit 3 - Flash Dwell	--	---	Yes / No	---	---
	Bit 4 - Reserved	--	---	---	---	---
	Bit 5 - Reserved	--	---	---	---	---
	Bit 6 - Reserved	--	---	---	---	---
	Bit 7 - Reserved	--	---	---	---	---
2.7.2.3	preemptLink	P	2.7 : M	Yes	0-255	
2.7.2.4	preemptDelay	P	2.7 : M	Yes	0-65535	
2.7.2.5	preemptMinimumDuration	P	2.7 : M	Yes	0-65535	
2.7.2.6	preemptMinimumGreen	P	2.7 : O	Yes / No	0-255	
2.7.2.7	preemptMinimumWalk	P	2.7 : O	Yes / No	0-255	
2.7.2.8	preemptEnterPedClear	P	2.7 : O	Yes	0-255	
2.7.2.9	preemptTrackGreen	P	2.7 : M	Yes	0-255	
2.7.2.10	preemptDwellGreen	P	2.7 : M	Yes	0-255	
2.7.2.11	preemptMaximumPresence	P	2.7 : M	Yes	0-65535	
2.7.2.12	preemptTrackPhase	P2	2.7 : M	Yes	string	
2.7.2.13	preemptDwellPhase	P2	2.7 : M	Yes	string	
2.7.2.14	preemptDwellPed	P2	2.7 : O	Yes	string	
2.7.2.15	preemptExitPhase	P2	2.7 : M	Yes	string	
2.7.2.16	preemptState	S	2.7 : O	Yes	1-9	
2.7.2.17	preemptTrackOverlap	P2	2.7 : M	Yes	string	
2.7.2.18	preemptDwellOverlap	P2	2.7 : M	Yes	string	
2.7.2.19	preemptCyclingPhase	P2	2.7 : M	Yes	string	
2.7.2.20	preemptCyclingPed	P2	2.7 : M	Yes	string	
2.7.2.21	preemptCyclingOverlap	P2	2.7 : M	Yes	string	
2.7.2.22	preemptEnterYellowChange	P	2.7 : M	Yes	0-255	
2.7.2.23	preemptEnterRedClear	P	2.7 : M	Yes	0-255	
2.7.2.24	preemptTrackYellowChange	P	2.7 : M	Yes	0-255	
2.7.2.25	preemptTrackRedClear	P	2.7 : M	Yes	0-255	
2.7.3	preemptControlTable	--	2.7 : O	Yes / No	---	---
	preemptControlEntry	--	2.7.3 : M	Yes	---	---
2.7.3.1	preemptControlNumber	S	2.7.3 : M	Yes	1-255	

PREEMPT CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.7.3.2	preemptControlState	C	2.7.3 : M	Yes	0-1	

A.11 RING CONFORMANCE GROUP

The Ring Conformance Group shall consist of the following objects:

RING CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.8	**Ring Conformance Group**	--	O	Yes / No	---	---
2.8.1	maxRings	S	2.8 : M	Yes	1-255	
2.8.2	maxSequences	S	2.8 : M	Yes	1-255	
2.8.3	sequenceTable	--	2.8 : M	Yes	---	---
	sequenceEntry	--	2.8 : M	Yes	---	---
2.8.3.1	sequenceNumber	S	2.8 : M	Yes	1-255	
2.8.3.2	sequenceRingNumber	S	2.8 : M	Yes	1-255	
2.8.3.3	sequenceData	P2	2.8 : M	Yes	string	
2.8.4	maxRingControlGroups	S	2.8 : M	Yes	1-255	
2.8.5	ringControlGroupTable	--	2.8 : M	Yes	---	---
	ringControlGroupEntry	--	2.8 : M	Yes	---	---
2.8.5.1	ringControlGroupNumber	S	2.8 : M	Yes	1-255	
2.8.5.2	ringControlGroupStopTime	C	2.8 : M	Yes	0-255	
2.8.5.3	ringControlGroupForceOff	C	2.8 : M	Yes	0-255	
2.8.5.4	ringControlGroupMax2	C	2.8 : O	Yes / No	0-255	
2.8.5.5	ringControlGroupMaxInhibit	C	2.8 : O	Yes / No	0-255	
2.8.5.6	ringControlGroupPedRecycle	C	2.8 : M	Yes	0-255	
2.8.5.7	ringControlGroupRedRest	C	2.8 : O	Yes / No	0-255	
2.8.5.8	ringControlGroupOmitRedClear	C	2.8 : O	Yes / No	0-255	
2.8.6	ringStatusTable	--	2.8 : O	Yes / No	---	---
	ringStatusEntry	--	2.8 : O	Yes / No	---	---
2.8.6.1	ringStatus	S	2.8 : O	Yes / No	0-255	

A.12 CHANNEL CONFORMANCE GROUP

The Channel Conformance Group shall consist of the following objects:

CHANNEL CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.9	**Channel Conformance Group**	--	O	Yes / No	----	---
2.9.1	maxChannels	S	2.9 : M	Yes	1-255	
2.9.2	channelTable	--	2.9 : M	Yes	---	---
	channelEntry	--	2.9 : M	Yes	---	---
2.9.2.1	channelNumber	S	2.9 : M	Yes	1-255	
2.9.2.2	channelControlSource	P	2.9 : M	Yes	0-255	
2.9.2.3	channelControlType	P	2.9 : M	Yes	1-4	
	other(1)	--	---	Yes / No	---	---
	phaseVehicle(2)	--	---	Yes / No	---	---
	phasePedestrian(3)	--	---	Yes / No	---	---
	overlap(4)	--	---	Yes / No	---	---
2.9.2.4	channelFlash	P	2.9 : M	Yes	0-255	
	Bit 0 - Reserved	--	---	---	---	---
	Bit 1 - Flash Yellow	--	---	Yes / No	---	---
	Bit 2 - Flash Red	--	---	Yes / No	---	---
	Bit 3 - Flash Alternate Half Hertz	--	---	Yes / No	---	---
	Bit 4 - Reserved	--	---	---	---	---
	Bit 5 - Reserved	--	---	---	---	---
	Bit 6 - Reserved	--	---	---	---	---
	Bit 7 - Reserved	--	---	---	---	---
2.9.2.5	channelDim	P	2.9 : M	Yes	0-255	
	Bit 0 - Dim Green	--	---	Yes / No	---	---
	Bit 1 - Dim Yellow	--	---	Yes / No	---	---
	Bit 2 - Dim Red	--	---	Yes / No	---	---
	Bit 3 - Dim Alternate Half Line Cycle	--	---	Yes / No	---	---
	Bit 4 - Reserved	--	---	---	---	---
	Bit 5 - Reserved	--	---	---	---	---
	Bit 6 - Reserved	--	---	---	---	---
	Bit 7 - Reserved	--	---	---	---	---
2.9.3	maxChannelStatusGroups	S	2.9 : M	Yes	1-255	
2.9.4	channelStatusGroupTable	--	2.9 : M	Yes	---	---
	channelStatusGroupEntry	--	2.9 : M	Yes	---	---
2.9.4.1	channelStatusGroupNumber	S	2.9 : M	Yes	1-255	
2.9.4.2	channelStatusGroupReds	S	2.9 : M	Yes	0-255	
2.9.4.3	channelStatusGroupYellows	S	2.9 : M	Yes	0-255	
2.9.4.4	channelStatusGroupGreens	S	2.9 : M	Yes	0-255	

A.13 OVERLAP CONFORMANCE GROUP

The Overlap Conformance Group shall consist of the following objects:

OVERLAP CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.10	**Overlap Conformance Group**	--	O	Yes / No	----	---
2.10.1	maxOverlaps	S	2.10 : M	Yes	1-255	
2.10.2	overlapTable	--	2.10 : M	Yes	---	---
	overlapEntry	--	2.10 : M	Yes	---	---
2.10.2.1	overlapNumber	S	2.10 : M	Yes	1-255	
2.10.2.2	overlapType	S	2.10 : M	Yes	1-3	
	other(1)	--	---	Yes / No	---	---
	normal(2)	--	---	Yes / No	---	---
	minusGreenYellow(3)	--	---	Yes / No	---	---
2.10.2.3	overlapIncludedPhases	P2	2.10 : M	Yes	string	
2.10.2.4	overlapModifierPhases	P2	2.10 : M	Yes	string	

OVERLAP CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.10.2.5	overlapTrailGreen	P	2.10 : M	Yes	0-255	
2.10.2.6	overlapTrailYellow	P	2.10 : M	Yes	0-255	
2.10.2.7	overlapTrailRed	P	2.10 : M	Yes	0-255	
2.10.3	maxOverlapStatusGroups	S	2.10 : M	Yes	1-255	
2.10.4	overlapStatusGroupTable	--	2.10 : M	Yes	---	---
	overlapStatusGroupEntry	--	2.10 : M	Yes	---	---
2.10.4.1	overlapStatusGroupNumber	S	2.10 : M	Yes	1-255	
2.10.4.2	overlapStatusGroupReds	S	2.10 : M	Yes	0-255	
2.10.4.3	overlapStatusGroupYellows	S	2.10 : M	Yes	0-255	
2.10.4.4	overlapStatusGroupGreens	S	2.10 : M	Yes	0-255	

A.14 TS 2 PORT 1 CONFORMANCE GROUP

The TS-2 Port 1 Conformance Group shall consist of the following objects:

TS 2 PORT 1 CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.11	**TS 2 PORT 1 CONFORMANCE GROUP**	--	O	Yes / No	----	---
2.11.1	maxPort1Addresses	S	2.11 : M	Yes	1-255	
2.11.2	port1Table	--	2.11 : M	Yes	---	---
	port1Entry	--	2.11 : M	Yes	---	---
2.11.2.1	port1Number	S	2.11 : M	Yes	1-255	
2.11.2.2	port1DevicePresent	P	2.11 : M	Yes	0-1	
2.11.2.3	port1Frame40Enable	P	2.11 : M	Yes	0-1	
2.11.2.4	port1Status	S	2.11 : M	Yes	1-3	
2.11.2.5	port1FaultFrame	S	2.11 : M	Yes	0-255	

A.15 BLOCK OBJECT CONFORMANCE GROUP

The Block Object Conformance Group shall consist of the following objects:

BLOCK OBJECT CONFORMANCE GROUP						
NTCIP 1202 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.12	**Block Object Conformance Group**	--	O	Yes / No	-----	----
2.12.1	ascBlockGetControl	C	2.12 : M	Yes	string	
2.12.2	ascBlockData	C	2.12 : M	Yes	string	
2.12.3	ascBlockErrorStatus	S	2.12 : M	Yes	0-65535	

NOTE—A device that supports this conformance group shall support the standard data block for each conformance group that is supported.

A.16 CONFIGURATION CONFORMANCE GROUP

The Configuration Conformance Group shall consist of the following objects:

CONFIGURATION CONFORMANCE GROUP						
NTCIP 1201 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.2	**Global Config Conformance Group**	--	M	Yes	-----	----
2.2.1	globalSetIDParmeter	S	2.2 : O	Yes / No	0-65535	
2.2.2	globalMaxModules	S	2.2 : M	Yes	1-255	
2.2.3	globalModuleTable	--	2.2 : M	Yes	---	---
	moduleTableEntry	--	2.2 : M	Yes	---	---
2.2.3.1	moduleNumber	S	2.2 : M	Yes	1-255	

CONFIGURATION CONFORMANCE GROUP						
NTCIP 1201 Clause	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
2.2.3.2	moduleDeviceNode	S	2.2 : M	Yes	OID	
2.2.3.3	moduleMake	S	2.2 : M	Yes	String	
2.2.3.4	moduleModel	S	2.2 : M	Yes	String	
2.2.3.5	moduleVersion	S	2.2 : M	Yes	String	
2.2.3.6	moduleType	S	2.2 : M	Yes	1-3	
	other(1)	--	---	Yes / No	---	---
	hardware(2)	--	---	Yes / No	---	---
	software(3)	--	---	Yes / No	---	---
2.2.4	controllerBaseStandards	S	2.2 : O	Yes / No	String	

A.17 DATABASE MANAGEMENT CONFORMANCE GROUP

The Database Management Conformance Group shall consist of the following objects:

DATABASE MANAGEMENT CONFORMANCE GROUP						
NTCIP 1201 Clause	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
2.3	**DB Management Conformance Group**	--	M	Yes	-----	----
2.3.1	dbCreateTransaction	C	M	Yes	1,2,3,6	
2.3.6	dbVerifyStatus	S	M	Yes	1-3	
2.3.7	dbVerifyError	S	M	Yes	String	

A.18 REPORT CONFORMANCE GROUP

The Report Conformance Group shall consist of the following objects:

REPORT CONFORMANCE GROUP						
NTCIP 1201 Clause	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
2.5	**Report Conformance Group**	--	O	Yes / No	-----	----
2.5.1	maxEventClasses	S	2.5 : M	Yes	1-255	
2.5.2	eventClassTable	--	2.5 : M	Yes	---	---
	eventClassEntry	--	2.5 : M	Yes	---	---
2.5.2.1	eventClassNumber	S	2.5 : M	Yes	1-255	
2.5.2.2	eventClassLimit	P	2.5 : M	Yes	0-255	
2.5.2.3	eventClassClearTime	P	2.5 : M	Yes	counter	
2.5.2.4	eventClassDescription	P	2.5 : O	Yes	string	
2.5.2.5	eventClassNumRowsInLog	S	2.5 : M	Yes	0-255	
2.5.2.6	eventClassNumEvents	S	2.5 : M	Yes	0-65535	
2.5.3	maxEventLogConfigs	S	2.5 : M	Yes	1-65535	
2.5.4	eventLogConfigTable	--	2.5 : M	Yes	---	---
	eventLogConfigEntry	--	2.5 : M	Yes	---	---
2.5.4.1	eventConfigID	S	2.5 : M	Yes	1-65535	
2.5.4.2	eventConfigClass	P	2.5 : M	Yes	1-255	
2.5.4.3	eventConfigMode	P	2.5 : M	Yes	1-7	
	other(1)	--	---	Yes / No	---	---
	onChange(2)	--	---	Yes / No	---	---
	greaterThanValue(3)	--	---	Yes / No	---	---
	smallerThanValue(4)	--	---	Yes / No	---	---
	hystersisBound(5)	--	---	Yes / No	---	---
	periodic(6)	--	---	Yes / No	---	---
	andedWithValue(7)	--	---	Yes / No	---	---
2.5.4.4	eventConfigCompareValue	P	2.5 : M	Yes	INT	
2.5.4.5	eventConfigCompareValue2	P	2.5 : M	Yes	INT	
2.5.4.6	eventConfigCompareOID	P	2.5 : M	Yes	OID	
2.5.4.7	eventConfigLogOID	P	2.5 : O	Yes	OID	
2.5.4.8	eventConfigAction	P	2.5 : M	Yes	1-3	
2.5.4.9	eventConfigStatus	S	2.5 : M	Yes	1-4	
2.5.5	maxEventLogSize	S	2.5 : M	Yes	1-65535	

REPORT CONFORMANCE GROUP						
NTCIP 1201 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.5.6	eventLogTable	--	2.5 : M	Yes	---	---
	eventLogEntry	--	2.5 : M	Yes	---	---
2.5.6.1	eventLogClass	S	2.5 : M	Yes	1-255	
2.5.6.2	eventLogNumber	S	2.5 : M	Yes	1-255	
2.5.6.3	eventLogID	S	2.5 : M	Yes	1-65535	
2.5.6.4	eventLogTime	S	2.5 : M	Yes	counter	
2.5.6.5	eventLogValue	S	2.5 : M	Yes	opaque (1)	
2.5.7	numEvents	S	2.5 : M	Yes	0-65535	

(1) For any NTCIP Compliant ASC implementation the value of the 'eventLogValue' OID has been changed from Opaque to Opaque (SIZE(0..40)).

A.19 AUXIO GROUP

The AUXIO Group shall consist of the following objects:

SFMP GROUP						
NTCIP 1201 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.8	**AUXIO GROUP**	--	X	No	----	---

A.20 PMPP GROUP

The PMPP Group shall consist of the following objects:

PMPP GROUP						
NTCIP 1201 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
2.6	**PMPP GROUP**	--	O	Yes / No	----	---
2.6.1	maxGroupAddress	S	2.6 : M	Yes	1..255	
2.6.2	hdlcGroupAddressTable	--	2.6 : M	Yes	---	---
	hdlcGroupAddressEntry	--	2.6 : M	Yes	---	---
2.6.2.1	hdlcGroupAddressIndex	S	2.6 : M	Yes	1..255	
2.6.2.3	hdlcGroupAddressNumber	P	2.6 : M	Yes	0-62	

A.21 SNMP GROUP

The SNMP Group shall consist of the following objects:

SNMP GROUP						
rfc 1213	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
snmp	**SNMP GROUP**	--	M	Yes	---	
snmp.1	snmpInPkts	S	snmp : M	Yes	Counter	
snmp.2	snmpOutPkts	S	snmp : M	Yes	Counter	
snmp.3	snmpInBadVersions	S	snmp : M	Yes	Counter	
snmp.4	snmpInBadCommunityNames	S	snmp : M	Yes	Counter	
snmp.5	snmpInBadCommunityUses	S	snmp : M	Yes	Counter	
snmp.6	snmpInASNParseErrs	S	snmp : M	Yes	Counter	
snmp.8	snmpInTooBigs	S	snmp : M	Yes	Counter	
snmp.9	snmpInNoSuchNames	S	snmp : M	Yes	Counter	
snmp.10	snmpInBadValues	S	snmp : M	Yes	Counter	
snmp.11	snmpInReadOnlys	S	snmp : M	Yes	Counter	
snmp.12	snmpInGenErrs	S	snmp : M	Yes	Counter	

SNMP GROUP						
rfc 1213	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
snmp.13	snmpInTotalReqVars	S	snmp : O	Yes / No	Counter	
snmp.14	snmpInTotalSetVars	S	snmp : O	Yes / No	Counter	
snmp.15	snmpInGetRequests	S	snmp : M	Yes	Counter	
snmp.16	snmpInGetNexts	S	snmp : M	Yes	Counter	
snmp.17	snmpInSetRequests	S	snmp : M	Yes	Counter	
snmp.18	snmpInGetResponses	S	snmp : M	Yes	Counter	
snmp.19	snmpInTraps	S	snmp : M	Yes	Counter	
snmp.20	snmpOutTooBigs	S	snmp : M	Yes	Counter	
snmp.21	snmpOutNoSuchNames	S	snmp : M	Yes	Counter	
snmp.22	snmpOutBadValues	S	snmp : M	Yes	Counter	
snmp.24	snmpOutGenErrs	S	snmp : M	Yes	Counter	
snmp.25	snmpOutGetRequests	S	snmp : M	Yes	Counter	
snmp.26	snmpOutGetNexts	S	snmp : M	Yes	Counter	
snmp.27	snmpOutSetRequests	S	snmp : M	Yes	Counter	
snmp.28	snmpOutGetResponses	S	snmp : M	Yes	Counter	
snmp.29	snmpOutTraps	S	snmp : O	Yes / No	Counter	
snmp.30	snmpEnableAuthenTraps	P	snmp : O	Yes / No	INT	

SNMP GROUP						
NTCIP 1103 Clause	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
A.3.1	snmpMaxPacketSize	S	snmp : M	Yes	484-65535	

A.22 SYSTEM GROUP

The System Group shall consist of the following objects:

SYSTEM GROUP						
rfc 1213	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
system	**SYSTEM GROUP**	--	M	Yes	---	
system 1	sysDescr	S	system : M	Yes	string	
system 2	sysObjectID	S	system : M	Yes	OID	
system 3	sysUpTime	S	system : M	Yes	TimeTicks	
system 4	sysContact	P	system : M	Yes	string	
system 5	sysName	P	system : M	Yes	string	
system 6	sysLocation	P	system : M	Yes	string	
system 7	sysServices	S	system : M	Yes	0..127	

A.23 SFMP GROUP

The SFMP Group shall consist of the following objects:

SFMP GROUP						
NTCIP 1103 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
A.4	**Objects for SFMP**	--	X	No	----	---

A.24 STMP GROUP

The STMP Group shall consist of the following objects:

STMP GROUP						
NTCIP 1103 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
A.5	**Objects for STMP**	--	O	Yes / No	---	---
A.5.1	Dynamic Object Definition	--	A.5 : M	Yes	---	---
	dynObjDef	--	A.5 : M	Yes	---	---
	dynObjEntry	--	A.5 : M	Yes	---	---
	dynObjNumber	S	A.5 : M	Yes	1..13	
	dynObjIndex	S	A.5 : M	Yes	1..255	
	dynObjVariable	C	A.5 : M	Yes	OID	
A.5.3	Dynamic Object Configuration	--	A.5 : M	Yes	---	---
	dynObjConfigTable	--	A.5 : M	Yes	---	---
	dynObjConfigEntry	--	A.5 : M	Yes	---	---
	dynObjConfigOwner	C	A.5 : M	Yes	string	
	dynObjConfigStatus	C	A.5 : M	Yes	1..3	
A.5.4	STMP Statistics	--	A.5 : M	Yes	--	
.1	stmpInPkts	S	A.5 : M	Yes	counter	
.2	stmpOutPkts	S	A.5 : M	Yes	counter	
.6	stmpInParseErrs	S	A.5 : M	Yes	counter	
.8	stmpInTooBigs	S	A.5 : M	Yes	counter	
.9	stmpInNoSuchNames	S	A.5 : M	Yes	counter	
.10	stmpInBadValues	S	A.5 : M	Yes	counter	
.11	stmpInReadOnlys	S	A.5 : M	Yes	counter	
.12	stmpInGenErrs	S	A.5 : M	Yes	counter	
.15	stmpInGetRequests	S	A.5 : M	Yes	counter	
.16	stmpInGetNexts	S	A.5 : M	Yes	counter	
.17	stmpInSetRequests	S	A.5 : M	Yes	counter	
.18	stmpInGetResponses	S	A.5 : M	Yes	counter	
.20	stmpOutTooBigs	S	A.5 : M	Yes	counter	
.21	stmpOutNoSuchNames	S	A.5 : M	Yes	counter	
.22	stmpOutBadValues	S	A.5 : M	Yes	counter	
.23	stmpOutReadOnly	S	A.5 : M	Yes	counter	
.24	stmpOutGenError	S	A.5 : M	Yes	counter	
.25	stmpOutGetRequests	S	A.5 : M	Yes	counter	
.26	stmpOutGetNexts	S	A.5 : M	Yes	counter	
.27	stmpOutSetRequests	S	A.5 : M	Yes	counter	
.28	stmpOutGetResponses	S	A.5 : M	Yes	counter	
.29	stmpOutTrapResponses	S	A.5 : M	Yes	counter	
.31	stmpInSetRequestsNoReply	S	A.5 : M	Yes	counter	
.32	stmpInSetResponses	S	A.5 : M	Yes	counter	
.33	stmpInErrorResponses	S	A.5 : M	Yes	counter	
.34	stmpOutSetRequestsNoReply	S	A.5 : M	Yes	counter	
.35	stmpOutSetResponses	S	A.5 : M	Yes	counter	
.35	stmpOutErrorResponses	S	A.5 : M	Yes	counter	
A.5.5	STMP Configuration	--	A.5 : M	Yes	---	---
.1	dynamicObjectPersistence	P	A.5 : M	Yes	0-65535	
.2	dynamicObjectTable-ConfigID	S	A.5 : M	Yes	0-65535	

A.25 LOGICAL NAME GROUP

The Logical Name Group shall consist of the following objects:

LOGICAL NAME GROUP						
NTCIP 1103 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
A.6	**Objects for Logical Names**	--	O	Yes / No	----	---
A.6.1	logicalNameTranslationTablemaxEntries	S	A.6 : M	Yes	1..255	
A.6.2	logicalNameTranslationTable	--	A.6 : M	Yes	---	---
	logicalNameTranslationEntry	--	A.6 : M	Yes	---	---
A.6.2.1	logicalNameTranslationindex	S	A.6 : M	Yes	INT	
A.6.2.2	logicalNameTranslationlogicalName	P	A.6 : M	Yes	String	
A.6.2.3	logicalNameTranslationnetworkAddress	P	A.6 : M	Yes	String	
A.6.2.4	logicalNameTranslationstatus	C	A.6 : M	Yes	INT	

A.26 TRAP MANAGEMENT GROUP

The Trap Management Group shall consist of the following objects:

TRAP MANAGEMENT GROUP						
NTCIP 1103 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
A.7	**Objects for Trap Management**	--	O	Yes / No	----	---

A.27 SECURITY GROUP

The Security Group shall consist of the following objects:

SECURITY GROUP						
NTCIP 1103 Clause	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
A.10	**SECURITY GROUP**	--	M	Yes	----	---
A.10.1	communityNameAdmin	P	A.10 : M	Yes	string	
A.10.2	communityNamesMax	S	A.10 : M	Yes	1..255	
A.10.3	communityNameTable	--	A.10 : M	Yes	---	---
	communityNameTableEntry	--	A.10 : M	Yes	---	---
A.10.3.1	communityNameIndex	S	A.10 : M	Yes	1..255	
A.10.3.2	communityNameUser	P	A.10 : M	Yes	string	
A.10.3.3	communityNameAccessMask	P	A.10 : M	Yes	gauge	

A.28 RS232 GROUP

The RS232 Group shall consist of the following objects:

RS232 GROUP						
rfc 1317	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
rs232	**RS232 GROUP**	--	O	Yes / No	----	---
rs232.1	rs232Number	S	rs232 : M	Yes	INT	
rs232.2	rs232PortTable	--	rs232 : M	Yes	---	---
	rs232PortEntry	--	rs232 : M	Yes	---	---
rs232.2.1	rs232PortIndex	S	rs232 : M	Yes	INT	
rs232.2.2	rs232PortType	S	rs232 : M	Yes	1..5	
	other(1)	--	---	Yes / No	---	---
	rs232(2)	--	---	Yes / No	---	---
	rs422(3)	--	---	Yes / No	---	---
	rs423(4)	--	---	Yes / No	---	---
	v35(5)	--	---	Yes / No	---	---
rs232.2.3	rs232PortInSigNumber	S	rs232 : O	Yes / No	INT	
rs232.2.4	rs232PortOutSigNumber	S	rs232 : O	Yes / No	INT	
rs232.2.5	rs232PortInSpeed	P	rs232 : M	Yes	INT	
rs232.2.6	rs232PortOutSpeed	P	rs232 : M	Yes	INT	
rs232.3	rs232AsyncPortTable	--	rs232 : M	Yes	---	---
	rs232AsyncPortEntry	--	rs232 : M	Yes	---	---
rs232.3.1	rs232AsyncPortIndex	S	rs232 : M	Yes	INT	
rs232.3.2	rs232AsyncPortBits	P	rs232 : O	Yes / No	5..8	
	five(5)	--	---	Yes / No	---	---
	six(6)	--	---	Yes / No	---	---
	seven(7)	--	---	Yes / No	---	---
	eight(8)	--	---	Yes / No	---	---
rs232.3.3	rs232AsyncPortStopBits	P	rs232 : O	Yes / No	1..4	
	one(1)	--	---	Yes / No	---	---
	two(2)	--	---	Yes / No	---	---
	one-and-half(3)	--	---	Yes / No	---	---
	dynamic(4)	--	---	Yes / No	---	---
rs232.3.4	rs232AsyncPortParity	P	rs232 : O	Yes / No	1..5	
	none(1)	--	---	Yes / No	---	---
	odd(2)	--	---	Yes / No	---	---
	even(3)	--	---	Yes / No	---	---
	mark(4)	--	---	Yes / No	---	---
	space(5)	--	---	Yes / No	---	---
rs232.3.5	rs232AsyncPortAutobaud	P	rs232 : O	Yes / No	1..2	
	enabled(1)	--	---	Yes / No	---	---
	disabled(2)	--	---	Yes / No	---	---
rs232.3.6	rs232AsyncPortParityErrs	S	rs232 : O	Yes / No	Counter	

RS232 GROUP						
rfc 1317	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
rs232.3.7	rs232AsyncPortFramingErrs	S	rs232 : M	Yes	Counter	
rs232.3.8	rs232AsyncPortOverrunErrs	S	rs232 : M	Yes	Counter	

A device may require the rs232PortInSpeed and rs232PortOutSpeed to be the same value. Therefore, a SET of rs232PortInSpeed may automatically SET rs232PortOutSpeed to the same value and vice-versa.

A.29 HDLC GROUP

The HDLC Group shall consist of the following objects:

HDLC GROUP						
rfc 1381	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
lapb	**HDLC GROUP**	--	O	Yes / No	----	---
lapb.1	lapbAdmnTable	--	lapb : M	Yes	---	---
	lapbAdmnEntry	--	lapb : M	Yes	---	---
lapb.1.1	lapbAdmnIndex	S	lapb : M	Yes	IfIndexType	
lapb.1.2	lapbAdmnStationType	P	lapb : O	Yes / No	1..3	
	dte(1)	--	---	Yes / No	---	---
	dce(2)	--	---	Yes / No	---	---
	dxe(3)	--	---	Yes / No	---	---
lapb.1.3	lapbAdmnControlField	P	lapb : O	Yes / No	1..2	
	modulo8(1)	--	---	Yes / No	---	---
	modulo128(2)	--	---	Yes / No	---	---
lapb.1.4	lapbAdmnTransmitN1FrameSize	P	lapb : M	Yes	P Integer	
lapb.1.5	lapbAdmnReceiveN1FrameSize	P	lapb : M	Yes	P Integer	
lapb.1.6	lapbAdmnTransmitKWindowSize	P	lapb : O	Yes / No	1..127	
lapb.1.7	lapbAdmnReceiveKWindowSize	P	lapb : O	Yes / No	1..127	
lapb.1.8	lapbAdmnN2RxmitCount	P	lapb : O	Yes / No	0..65535	
lapb.1.9	lapbAdmnT1AckTimer	P	lapb : M	Yes	P Integer	
lapb.1.10	lapbAdmnT2AckDelayTimer	P	lapb : M	Yes	P Integer	
lapb.1.11	lapbAdmnT3DisconnectTimer	P	lapb : M	Yes	P Integer	
lapb.1.12	lapbAdmnT4IdleTimer	P	lapb : M	Yes	P Integer	
lapb.1.13	lapbAdmnActionInitiate	P	lapb : O	Yes / No	1..5	
	sendSABM(1)	--	---	Yes / No	---	---
	sendDISC(2)	--	---	Yes / No	---	---
	sendDM(3)	--	---	Yes / No	---	---
	none(4)	--	---	Yes / No	---	---
	other(5)	--	---	Yes / No	---	---
lapb.1.14	lapbAdmnActionRecvDM	P	lapb : O	Yes / No	1..3	
	sendSABM(1)	--	---	Yes / No	---	---
	sendDISC(2)	--	---	Yes / No	---	---
	other(3)	--	---	Yes / No	---	---
lapb.2	lapbOperTable	--	lapb : M	Yes	----	---
	lapbOperEntry	--	lapb : M	Yes	----	---
lapb.2.1	lapbOperIndex	S	lapb : M	Yes	IfIndexType	
lapb.2.2	lapbOperStationType	S	lapb : O	Yes / No	1..3	
	dte(1)	--	---	Yes / No	---	---
	dce(2)	--	---	Yes / No	---	---
	dxe(3)	--	---	Yes / No	---	---
lapb.2.3	lapbOperControlField	S	lapb : O	Yes / No	1..2	
	modulo8(1)	--	---	Yes / No	---	---
	modulo128(2)	--	---	Yes / No	---	---
lapb.2.4	lapbOperTransmitN1FrameSize	S	lapb : O	Yes / No	P Integer	
lapb.2.5	lapbOperReceiveN1FrameSize	S	lapb : O	Yes / No	P Integer	
lapb.2.6	lapbOperTransmitKWindowSize	S	lapb : O	Yes / No	1..127	
lapb.2.7	lapbOperReceiveKWindowSize	S	lapb : O	Yes / No	1..127	
lapb.2.8	lapbOperN2RxmitCount	S	lapb : O	Yes / No	0..65535	
lapb.2.9	lapbOperT1AckTimer	S	lapb : O	Yes / No	P Integer	
lapb.2.10	lapbOperT2AckDelayTimer	S	lapb : O	Yes / No	P Integer	
lapb.2.11	lapbOperT3DisconnectTimer	S	lapb : O	Yes / No	P Integer	

HDLC GROUP						
rfc 1381	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
lapb.2.12	lapbOperT4IdleTimer	S	lapb : O	Yes / No	P Integer	
lapb.2.13	lapbOperPortId	S	lapb : M	Yes	OID	
lapb.2.14	lapbOperProtocolVersionID	S	lapb : O	Yes / No	OID	

'P Integer = Positive Integer

A.30 INTERFACES GROUP

The Interfaces Group shall consist of the following objects:

INTERFACES GROUP						
rfc 1213	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
if	**INTERFACES GROUP**	--	O	Yes / No	----	---
if.1	ifNumber	S	if : M	Yes	---	
if.2	ifTable	--	if : M	Yes	---	---
	ifEntry	--	if : M	Yes	---	---
if.2.1	ifIndex	S	if : M	Yes	INT	
if.2.2	ifDescr	S	if : M	Yes	string	
if.2.3	ifType	S	if : M	Yes	INT	
if.2.4	ifMtu	S	if : M	Yes	INT	
if.2.5	ifSpeed	S	if : M	Yes	gauge	
if.2.6	ifPhysAddress	S	if : M	Yes	PhysAddress	
if.2.7	ifAdminStatus	C	if : O	Yes / No	1-3	
if.2.8	ifOperStatus	S	if : M	Yes	1-3	
if.2.9	ifLastChange	S	if : O	Yes / No	TimeTicks	
if.2.10	ifInOctets	S	if : O	Yes / No	counter	
if.2.11	ifInUcastPkts	S	if : O	Yes / No	counter	
if.2.12	ifInNUcastPkts	S	if : O	Yes / No	counter	
if.2.13	ifInDiscards	S	if : O	Yes / No	counter	
if.2.14	ifInErrors	S	if : O	Yes / No	counter	
if.2.15	ifInUnknownProtos	S	if : O	Yes / No	counter	
if.2.16	ifOutOctets	S	if : O	Yes / No	counter	
if.2.17	ifOutUcastPkts	S	if : O	Yes / No	counter	
if.2.18	ifOutNUcastPkts	S	if : O	Yes / No	counter	
if.2.19	ifOutDiscards	S	if : O	Yes / No	counter	
if.2.20	ifOutErrors	S	if : O	Yes / No	counter	
if.2.21	ifOutQLen	S	if : O	Yes / No	gauge	
if.2.22	ifSpecific	S	if : O	Yes / No	OID	

A.31 IP GROUP

The IP Group shall consist of the following objects:

IP GROUP						
rfc 1213	Object Name	Object Type	Object Status	Object Support	Allowed Values	Supported Values
ip	**IP GROUP**	--	O	Yes / No	----	---
ip.1	ipForwarding	C	ip : M	Yes	1-2	
ip.2	ipDefaultTTL	C	ip : M	Yes	INT	
ip.3	ipInReceives	S	ip : M	Yes	counter	
ip.4	ipInHdrErrors	S	ip : M	Yes	counter	
ip.5	ipInAddrErrors	S	ip : M	Yes	counter	
ip.6	ipForwDatagrams	S	ip : M	Yes	counter	
ip.7	ipInUnknownProtos	S	ip : M	Yes	counter	
ip.8	ipInDiscards	S	ip : M	Yes	counter	
ip.9	ipInDelivers	S	ip : M	Yes	counter	
ip.10	ipOutRequests	S	ip : M	Yes	counter	

IP GROUP						
rfc 1213	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
ip.11	ipOutDiscards	S	ip : M	Yes	counter	
ip.12	ipOutNoRoutes	S	ip : M	Yes	counter	
ip.13	ipReasmTimeout	S	ip : M	Yes	counter	
ip.14	ipReasmReqds	S	ip : M	Yes	counter	
ip.15	ipReasmOKs	S	ip : M	Yes	counter	
ip.16	ipReasmFails	S	ip : M	Yes	counter	
ip.17	ipFragOKs	S	ip : M	Yes	counter	
ip.18	ipFragFails	S	ip : M	Yes	counter	
ip.19	ipFragCreates	S	ip : M	Yes	counter	
ip.20	ipAddrTable	--	ip : M	Yes	---	
ip.20.1	ipAddrEntry	--	ip : M	Yes	---	
ip.20.1.1	ipAdEntAddr	S	ip : M	Yes	IpAddress	
ip.20.1.2	ipAdEntIfIndex	S	ip : M	Yes	INT	
ip.20.1.3	ipAdEntNetMask	S	ip : M	Yes	IpAddress	
ip.20.1.4	ipAdEntBcastAddr	S	ip : M	Yes	INT	
ip.20.1.5	ipAdEntReasmMaxSize	S	ip : M	Yes	0-65535	
ip.21	ipRouteTable	--	ip : M	Yes	---	
ip.21.1	ipRouteEntry	--	ip : M	Yes	---	
ip.21.1.1	ipRouteDest	C	ip : M	Yes	IpAddress	
ip.21.1.2	ipRouteIfIndex	C	ip : M	Yes	INT	
ip.21.1.3	ipRouteMetric1	C	ip : M	Yes	INT	
ip.21.1.4	ipRouteMetric2	C	ip : M	Yes	INT	
ip.21.1.5	ipRouteMetric3	C	ip : M	Yes	INT	
ip.21.1.6	ipRouteMetric4	C	ip : M	Yes	INT	
ip.21.1.7	ipRouteNextHop	C	ip : M	Yes	IpAddress	
ip.21.1.8	ipRouteType	C	ip : M	Yes	1-4	
ip.21.1.9	ipRouteProto	S	ip : M	Yes	1-14	
ip.21.1.10	ipRouteAge	C	ip : M	Yes	INT	
ip.21.1.11	ipRouteMask	C	ip : M	Yes	IpAddress	
ip.21.1.12	ipRouteMetric5	C	ip : M	Yes	INT	
ip.21.1.13	ipRouteInfo	S	ip : M	Yes	OID	
ip.22	ipNetToMediaTable	--	ip : M	Yes	---	
ip.22.1	ipNetToMediaEntry	--	ip : M	Yes	---	
ip.22.1.1	ipNetToMediaIfIndex	C	ip : M	Yes	INT	
ip.22.1.2	ipNetToMediaPhysAddress	C	ip : M	Yes	PhysAddress	
ip.22.1.3	ipNetToMediaNetAddress	C	ip : M	Yes	IpAddress	
ip.22.1.4	ipNetToMediaType	C	ip : M	Yes	1-4	
ip.23	ipRoutingDiscards	S	ip : M	Yes	counter	

A.32 ICMP GROUP

The ICMP Group shall consist of the following objects:

ICMP GROUP						
rfc 1213	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
icmp	**ICMP GROUP**	--	O	Yes / No	----	---
icmp.1	icmpInMsgs	S	icmp : M	Yes	counter	
icmp.2	icmpInErrors	S	icmp : M	Yes	counter	
icmp.3	icmpInDestUnreachs	S	icmp : M	Yes	counter	
icmp.4	icmpInTimeExcds	S	icmp : M	Yes	counter	
icmp.5	icmpInParmProbs	S	icmp : M	Yes	counter	
icmp.6	icmpInSrcQuenchs	S	icmp : M	Yes	counter	
icmp.7	icmpInRedirects	S	icmp : M	Yes	counter	
icmp.8	icmpInEchos	S	icmp : M	Yes	counter	
icmp.9	icmpInEchoReps	S	icmp : M	Yes	counter	
icmp.10	icmpInTimestamps	S	icmp : M	Yes	counter	
icmp.11	icmpInTimestampReps	S	icmp : M	Yes	counter	
icmp.12	icmpInAddrMasks	S	icmp : M	Yes	counter	
icmp.13	icmpInAddrMaskReps	S	icmp : M	Yes	counter	
icmp.14	icmpOutMsgs	S	icmp : M	Yes	counter	
icmp.15	icmpOutErrors	S	icmp : M	Yes	counter	
icmp.16	icmpOutDestUnreachs	S	icmp : M	Yes	counter	
icmp.17	icmpOutTimeExcds	S	icmp : M	Yes	counter	
icmp.18	icmpOutParmProbs	S	icmp : M	Yes	counter	
icmp.19	icmpOutSrcQuenchs	S	icmp : M	Yes	counter	
icmp.20	icmpOutRedirects	S	icmp : M	Yes	counter	
icmp.21	icmpOutEchos	S	icmp : M	Yes	counter	
icmp.22	icmpOutEchoReps	S	icmp : M	Yes	counter	
icmp.23	icmpOutTimestamps	S	icmp : M	Yes	counter	
icmp.24	icmpOutTimestampReps	S	icmp : M	Yes	counter	
icmp.25	icmpOutAddrMasks	S	icmp : M	Yes	counter	
icmp.26	icmpOutAddrMaskReps	S	icmp : M	Yes	counter	

A.33 TCP GROUP

The TCP Group shall consist of the following objects:

TCP GROUP						
rfc 1213	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
tcp	**TCP GROUP**	--	O	Yes / No	----	---
tcp.1	tcpRtoAlgorithm	S	tcp : M	Yes	1-4	
tcp.2	tcpRtoMin	S	tcp : M	Yes	INT	
tcp.3	tcpRtoMax	S	tcp : M	Yes	INT	
tcp.4	tcpMaxConn	S	tcp : M	Yes	INT	
tcp.5	tcpActiveOpens	S	tcp : M	Yes	counter	
tcp.6	tcpPassiveOpens	S	tcp : M	Yes	counter	
tcp.7	tcpAttemptFails	S	tcp : M	Yes	counter	
tcp.8	tcpEstabResets	S	tcp : M	Yes	counter	
tcp.9	tcpCurrEstab	S	tcp : M	Yes	gauge	
tcp.10	tcpInSegs	S	tcp : M	Yes	counter	
tcp.11	tcpOutSegs	S	tcp : M	Yes	counter	
tcp.12	tcpRetransSegs	S	tcp : M	Yes	counter	
tcp.13	tcpConnTable	--	tcp : M	Yes	---	
tcp.13.1	tcpConnEntry	--	tcp : M	Yes	---	
tcp.13.1.1	tcpConnState	C	tcp : M	Yes	1-12	
tcp.13.1.2	tcpConnLocalAddress	S	tcp : M	Yes	IpAddress	
tcp.13.1.3	tcpConnLocalPort	S	tcp : M	Yes	0-65535	
tcp.13.1.4	tcpConnRemAddress	S	tcp : M	Yes	IpAddress	
tcp.13.1.5	tcpConnRemPort	S	tcp : M	Yes	0-65535	

TCP GROUP						
rfc 1213	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
tcp.14	tcpInErrs	S	tcp : M	Yes	counter	
tcp.15	tcpOutRsts	S	tcp : M	Yes	counter	

A.34 UDP GROUP

The UDP Group shall consist of the following objects:

UDP GROUP						
rfc 1213	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
udp	**UDP GROUP**	--	O	Yes / No	----	---
udp.1	udpInDatagrams	S	udp : M	Yes	INT	
udp.2	udpNoPorts	S	udp : M	Yes	counter	
udp.3	udpInErrors	S	udp : M	Yes	counter	
udp.4	udpOutDatagrams	S	udp : M	Yes	counter	
udp.5	udpTable	--	udp : M	Yes	---	
udp.5.1	udpEntry	--	udp : M	Yes	---	
udp.5.1.1	udpLocalAddress	S	udp : M	Yes	IpAddress	
udp.5.1.2	udpLocalPort	S	udp : M	Yes	0-65535	

A.35 ETHERNET GROUP

The Ethernet Group shall consist of the following objects:

ETHERNET GROUP						
rfc 1643	**Object Name**	**Object Type**	**Object Status**	**Object Support**	**Allowed Values**	**Supported Values**
dot3	**ETHERNET GROUP**	--	O	Yes / No	----	---
dot3.2	dot3StatsTable	--	dot3 : M	Yes	---	---
dot3.2.1	dot3StatsEntry	--	dot3 : M	Yes	---	---
dot3.2.1.1	dot3StatsIndex	S	dot3 : M	Yes	INT	
dot3.2.1.2	dot3StatsAlignmentErrors	S	dot3 : M	Yes	counter	
dot3.2.1.3	dot3StatsFCSErrors	S	dot3 : M	Yes	counter	
dot3.2.1.4	dot3StatsSingleCollisionFrames	S	dot3 : M	Yes	counter	
dot3.2.1.5	dot3StatsMultipleCollisionFrames	S	dot3 : M	Yes	counter	
dot3.2.1.6	dot3StatsSQETestErrors	S	dot3 : M	Yes	counter	
dot3.2.1.7	dot3StatsDeferredTransmissions	S	dot3 : M	Yes	counter	
dot3.2.1.8	dot3StatsLateCollisions	S	dot3 : M	Yes	counter	
dot3.2.1.9	dot3StatsExcessiveCollisions	S	dot3 : M	Yes	counter	
dot3.2.1.10	dot3StatsInternalMacTransmitErrors	S	dot3 : M	Yes	counter	
dot3.2.1.11	dot3StatsCarrierSenseErrors	S	dot3 : M	Yes	counter	
dot3.2.1.13	dot3StatsFrameTooLongs	S	dot3 : M	Yes	counter	
dot3.2.1.16	dot3StatsInternalMacReceiveErrors	S	dot3 : M	Yes	counter	
dot3.2.1.17	dot3StatsEtherChipSet	S	dot3 : M	Yes	OID	
dot3.5	dot3CollTable	--	dot3 : O	Yes / No		
dot3.5.1	dot3CollEntry	--	dot3 : O	Yes / No		
dot3.5.1.2	dot3CollCount	S	dot3 : O	Yes / No	INT	
dot3.5.1.3	dot3CollFrequencies	S	dot3 : O	Yes / No	counter	
dot3.6	dot3Tests	--	dot3 : O	Yes / No		
dot3.6.1	dot3TestTdr	S	dot3 : O	Yes / No		
dot3.6.2	dot3TestLoopBack	S	dot3 : O	Yes / No		
dot3.7	dot3Errors	--	dot3 : O	Yes / No		
dot3.7.1	dot3ErrorInitError	S	dot3 : O	Yes / No		
dot3.7.2	dot3ErrorLoopbackError	S	dot3 : O	Yes / No		

Annex B
CONSISTENCY CHECKS
(Normative)

Consistency checks assure that certain critical objects are checked “in context” and treated as interrelated values rather than separate non-related data items.

When data is downloaded to a CU operating in the “transaction” mode, as defined by the dbCreateTransaction object defined in NTCIP 1201, consistency checks shall be performed on downloaded data when the “verify” state is commanded. The consistency checks that shall occur and corresponding error messages are described below. Error messages, if any, may be examined by reading the dbTransactionError object once the CU has entered the “done” mode.

B.1 CONSISTENCY CHECK RULES

The consistency check rule is stated first, followed by the corresponding error message(s).

- Concurrent Phases, as defined by the phaseConcurrency object, must be in a different ring from phaseNumber (phase being defined). The error message indicates one or more defined concurrent phases have the same ring assignment as phaseNumber. The value “xx” corresponds to phaseNumber.

  “PHASE xx CONCURRENCY FAULT”

  An example: phaseConcurrency.1 (Phase 1 concurrent phases) includes Phase 2 and phaseRing.1 (Phase 1 Ring) equals phaseRing.2 (Phase 2 Ring). An error message of "PHASE 01 CONCURRENCY FAULT" is provided.

- Concurrent Phases, as defined by the phaseConcurrency object, must be mutually concurrent with phaseNumber (phase being defined). The error message indicates one or more defined concurrent phases does not include phaseNumber as a concurrent phase. The value "xx" corresponds to phaseNumber.

  "PHASE xx MUTUAL FAULT"

  An example: phaseConcurrency.1 (Phase 1 concurrent phases) includes phase 5 and phaseConcurrency.5 (Phase 5 concurrent phases) does not include phase 1. An error message of "PHASE 01 MUTUAL FAULT" is provided.

- Phase Sequences, as defined by the sequenceData object, must include phases only once in a given phase sequence. The error message indicates a phase appears more than once in a phase sequence. The value “xx” corresponds to sequenceNumber for sequenceData.

  “SEQ xx SAME PHASE FAULT”

  An example: sequenceData.1.1 (Sequence 01 / Ring 1) is 01-02-03-04-01 (Phase 1 appears twice). An error message of "SEQ 01 SAME PHASE FAULT" is provided.

- Phase Sequences, as defined by the sequenceData object, must include only phases with a ring assignment (phaseRing) equal to sequenceRingNumber. The error message indicates a phase defined by sequenceData does not have a phaseRing equal to sequenceRingNumber. The value “xx” corresponds to sequenceNumber. The value "#" corresponds to sequenceRingNumber.

  “SEQ xx RING # FAULT”

  An example: sequenceData.1.1 (Sequence 01 / Ring 1) is 01-02-03-04-05 and all phaseRing parameters = 1 except phaseRing.5 = 2. An error message of "SEQ 01 RING 1 FAULT" is provided.

- Phase Sequences, as defined by the sequenceData object, must include all phases with a ring assignment (phaseRing) equal to sequenceRingNumber. The error message indicates a phase has been omitted in the sequenceData for sequenceRingNumber. The value “xx” corresponds to sequenceNumber. The value "#" corresponds to sequenceRingNumber.

  “SEQ xx RING # PHS OMITTED”

  An standard dual ring example: sequenceData.1.1 (Sequence 01 / Ring 1) is 01-02-03 (does not include Phase 4). An error message of "SEQ 01 RING 1 PHS OMITTED" is provided.

- Phase Sequences, as defined by the sequenceData object, must be ordered such that all sequenceRingNumber phases within a Concurrency Group can be serviced sequentially without leaving the Concurrency Group of which they are a member. The error message indicates all phases in a Concurrency Group could not be serviced sequentially. The value “xx” corresponds to sequenceNumber.

  “SEQ xx RING SEQ FAULT”

  An standard dual ring example: sequenceData.1.1 (Sequence 01 / Ring 1) is 01-03-02-04 and sequenceData.1.2 (Sequence 01 / Ring 2) is 05-06-07-08. An error message of "SEQ 01 RING SEQ FAULT" is provided.

- Phase Sequences, as defined by the sequenceData object; phases must be arranged so Concurrency Groups of which phases are a member are sequenced in the same order in all rings for a given sequenceNumber. The error message indicates Concurrency Groups are not in the same order for all. The value “xx” corresponds to sequenceNumber.

  “SEQ xx CG SEQ FAULT”

  An standard dual ring example: sequenceData.1.1 (Sequence 01 / Ring 1) is 01-02-03-04 and sequenceData.1.2 (Sequence 01 / Ring 2) is 07-08-05-06. An error message of "SEQ 01 CG SEQ FAULT" is provided.

- Phase Sequences, as defined by the sequenceData object; phases must be arranged so that it is possible to service all phases (not skip any phase due to compatibility constraints) in all rings in the order defined.

  "**SEQ xx SEQUENCING FAULT**"

  An example (lead-lag dual ring where phase 1 & 5 can not operate concurrently): sequenceData.1.1 (Sequence 01 / Ring 1) is 02-01-03-04 and sequenceData.1.2 (Sequence 01 / Ring 2) is 06-05-07-08. An error message of "SEQ 01 SEQUENCING FAULT" is provided.

- The following objects define functionality related to phase assignments. Consistency checks among other things, insure that phases specified by these objects may operate concurrently and are defined only once in each string parameter. Note that if the objects are not defined, operation between different CU's may be inconsistent.

  Phase Startup (phaseStartup)
  Automatic Flash Entry Phases (phaseOptions[1])
  Automatic Flash Exit Phases (phaseOptions[2])
  Overlap Included Phases (overlapIncludedPhases)
  Overlap Modifier Phases (overlapModifierPhases)
  Preempt Track Clear Phases (preemptTrackPhase)
  Preempt Dwell Phases (preemptDwellPhase)
  Preempt Dwell Peds (preemptDwellPed)
  Preempt Exit Phases (preemptExitPhase)
  Preempt Cycling Phases (preemptCyclingPhase)
  Preempt Cycling Ped (preemptCyclingPed)

  When the defined phases CAN NOT time concurrently:
  "START PHASE CG FAULT"
  "FLASH ENTRY CG FAULT"
  "FLASH EXIT CG FAULT"
  "PE TRACK PHASE CG FAULT"
  "PE DWELL PHASE CG FAULT"
  "PE EXIT PHASE CG FAULT"

  When the defined phases are in the same ring:
  "START PHASE RING FAULT"
  "FLASH ENTRY RING FAULT"
  "FLASH EXIT RING FAULT"
  "PE TRACK PHASE RING FAULT"
  "PE DWELL PHASE RING FAULT"
  "PE EXIT PHASE RING FAULT"

  When the defined phases are in the string parameter more than once:
  "OVLP INC PHASE MULTI FAULT"
  "OVLP MOD PHASE MULTI FAULT"
  "PE TRACK PHASE MULTI FAULT"
  "PE DWELL PHASE MULTI FAULT"
  "PE DWELL PED MULTI FAULT"
  "PE EXIT PHASE MULTI FAULT"
  "PE CYCLING PHASE MULTI FAULT"
  "PE CYCLING PED MULTI FAULT"

  When a defined phase is disabled:
  "START PHASE DISABLE FAULT"
  "FLASH ENTRY DISABLE FAULT"
  "FLASH EXIT DISABLE FAULT"
  "PE TRACK PHASE DISABLE FAULT"
  "PE DWELL PHASE DISABLE FAULT"
  "PE EXIT PHASE DISABLE FAULT"

  When a peds parent phase is NOT active:
  "PE DWELL PED PARENT FAULT"

- The following objects define functionality related to overlap assignments. Consistency checks insure that overlaps specified by these objects may only be active when an included phase (overlapIncludedPhases) is active.

Preempt Track Clear Overlaps (preemptTrackOverlap)
Preempt Dwell Overlaps (preemptDwellOverlap)
Preempt Cycling Overlap (preemptCyclingOverlap)

When an included phase IS NOT defined to be active:
"PE TRACK OVERLAP FAULT"
"PE DWELL OVERLAP FAULT"

When the defined overlaps are in the string parameter more than once:
"PE TRACK OVLP MULTI FAULT"
"PE DWELL OVLP MULTI FAULT"
"PE CYCLING OVLP MULTI FAULT"

- When no consistency faults are detected in the data when leaving "transaction" mode, the following shall be written to the dbVerifyError object:

"NO VERIFICATION ERROR"

Note that the order of the checks is not defined. Therefore, for a given set of 'bad' data, the Error Message between different CU's may be inconsistent.

B.2 MANUFACTURER SPECIFIC CONSISTENCY CHECKS

There are functional differences between CU's manufactured by different vendors. It is assumed that manufacturers will use consistency checks, beyond those specified here, to prevent accidental corruption of the CU database. Any such consistency checks must utilize the error reporting mechanism defined by this standard. These consistency checks and associated error messages should be clearly described and documented.

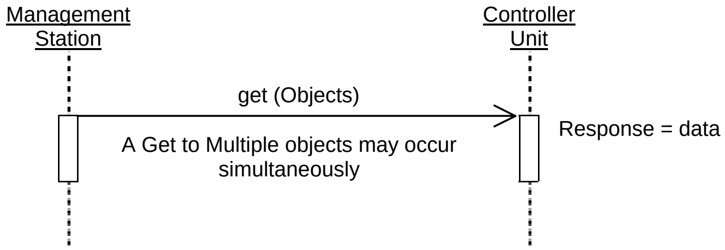
Annex C
CONCEPT OF OPERATIONS
(Informative)

This Annex provides:

1. Examples of how a management station may interface with an ASC complying with this standard as envisioned by the authors. Any ASC claiming conformance with the subject features depicted in these figures shall support the exchanges as shown. However, the flexible design of the NTCIP protocols allows a large number of other possibilities and these figures do not limit any other requirements of these standards. These diagrams are merely provided to promote a common understanding of how systems may be designed in order to increase the likelihood of interoperability in deployed systems.
2. Supplemental information on overlap sequences based on programming data for 'overlapIncludedPhases' and 'overlapModifierPhases'.

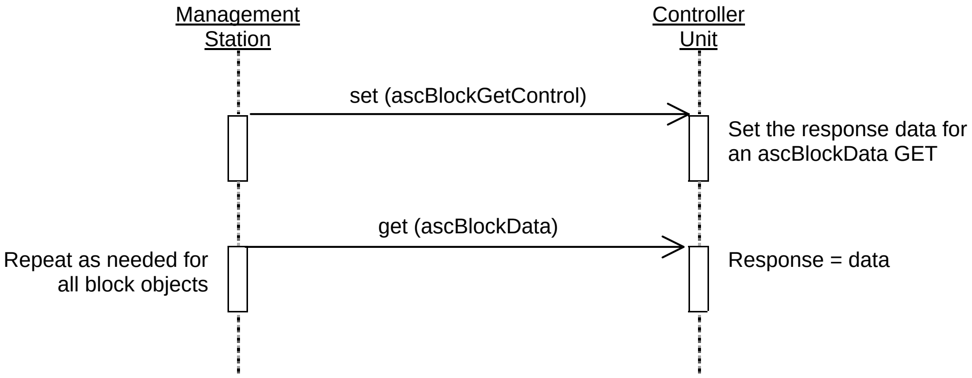
C.1 GET TYPE 'C' - 'P' - 'S' OBJECTS

This use case applies when getting Type 'C' - 'P' - 'S' objects.



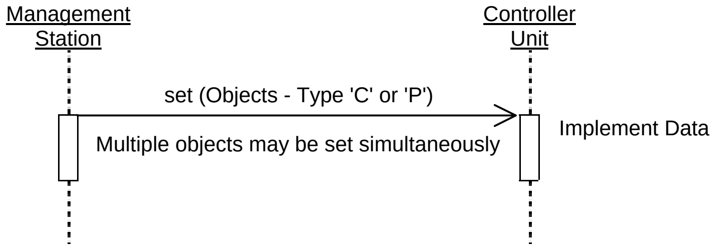
C.2 GET BLOCK DATA

This use shall applies when getting block data via object 'ascBlockData'.



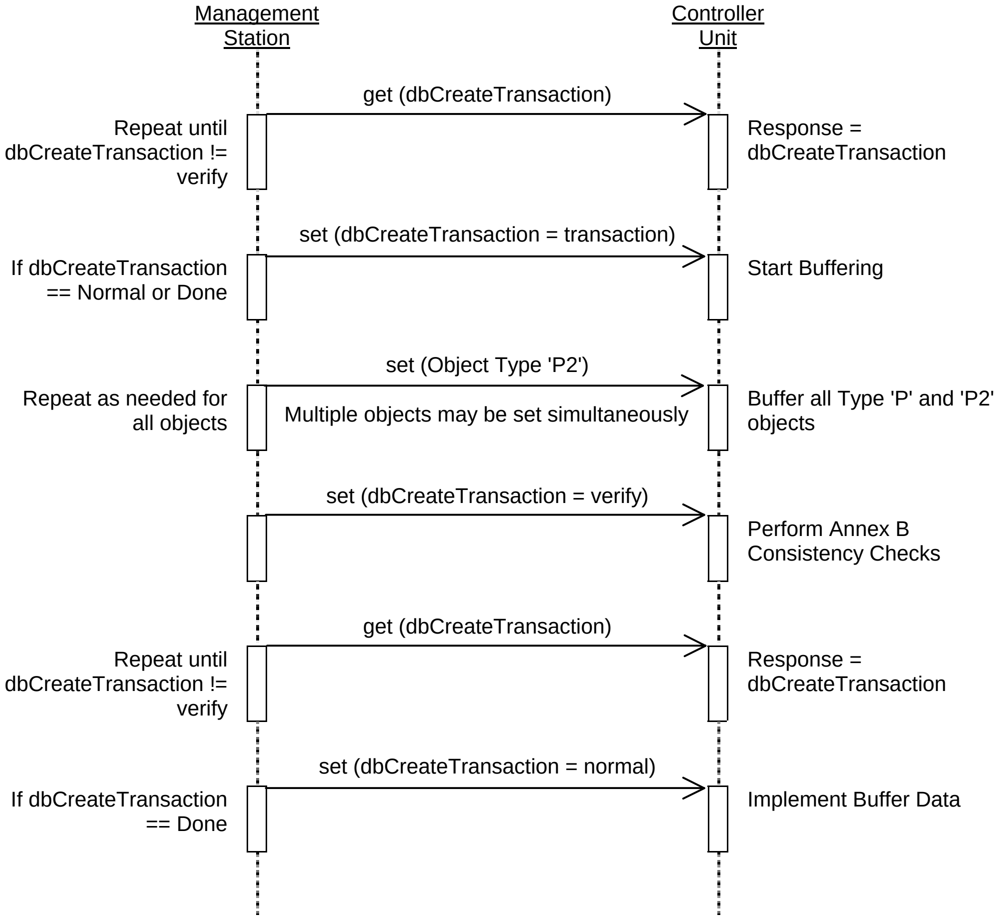
C.3 SET TYPE 'C' OR 'P' OBJECTS

This use case applies when only Type 'C' or 'P' objects are included in the data to be set.


Management Station
Controller Unit
set (Objects - Type 'C' or 'P')
Multiple objects may be set simultaneously
Implement Data

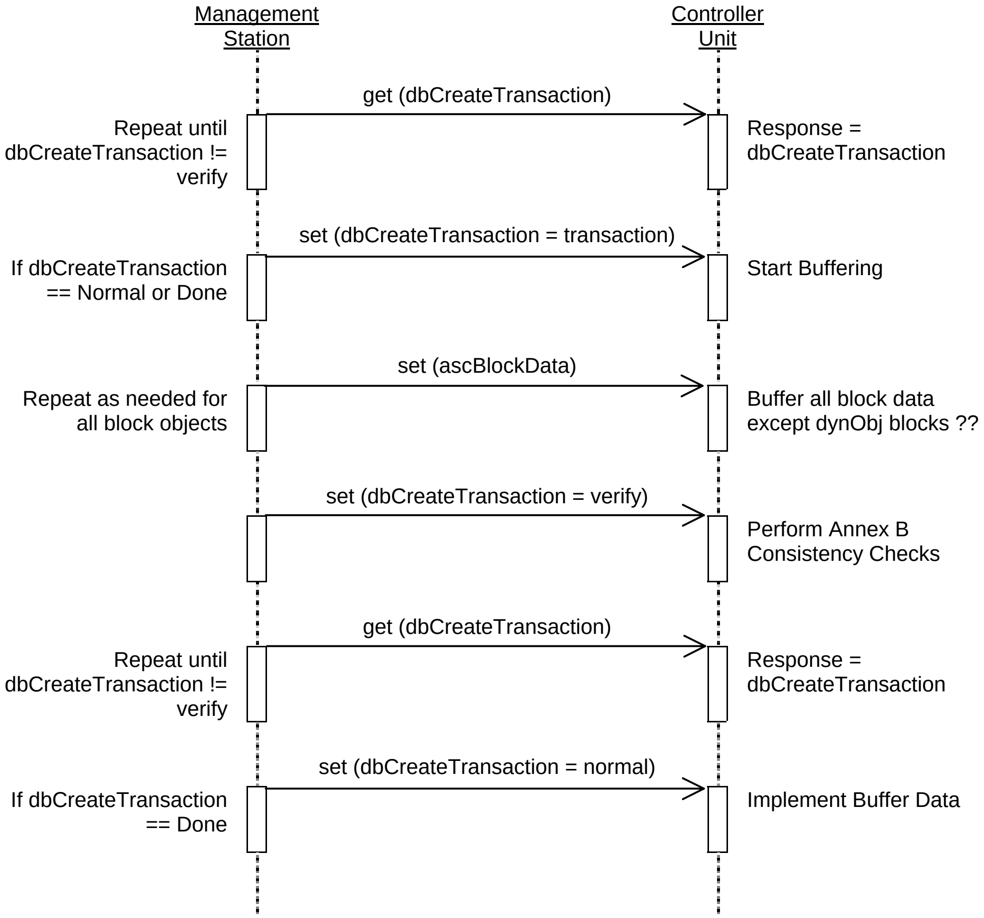
C.4 SET TYPE 'P2' OBJECTS

This use case applies when Type 'P2' objects are included in the data to be set.


Management Station
Controller Unit
get (dbCreateTransaction)
Repeat until dbCreateTransaction != verify
Response = dbCreateTransaction
set (dbCreateTransaction = transaction)
If dbCreateTransaction == Normal or Done
Start Buffering
set (Object Type 'P2')
Repeat as needed for all objects
Multiple objects may be set simultaneously
Buffer all Type 'P' and 'P2' objects
set (dbCreateTransaction = verify)
Perform Annex B Consistency Checks
get (dbCreateTransaction)
Repeat until dbCreateTransaction != verify
Response = dbCreateTransaction
set (dbCreateTransaction = normal)
If dbCreateTransaction == Done
Implement Buffer Data

C.5 SET BLOCK DATA

This use case applies when setting block data via object 'ascBlockData'.


Management Station
Controller Unit
get (dbCreateTransaction)
Repeat until dbCreateTransaction != verify
Response = dbCreateTransaction
set (dbCreateTransaction = transaction)
If dbCreateTransaction == Normal or Done
Start Buffering
set (ascBlockData)
Repeat as needed for all block objects
Buffer all block data except dynObj blocks ??
set (dbCreateTransaction = verify)
Perform Annex B Consistency Checks
get (dbCreateTransaction)
Repeat until dbCreateTransaction != verify
Response = dbCreateTransaction
set (dbCreateTransaction = normal)
If dbCreateTransaction == Done
Implement Buffer Data

C.6 OVERLAP SUPPLEMENTAL

This clause provides supplemental information on overlap sequences based on programming data for 'overlapIncludedPhases' and 'overlapModifierPhases'.

Sample Overlap Programming Data:

overlapNumber	overlapType	overlapIncludedPhases	overlapModifierPhases
1	2	1 - 2	-
2	3	1 - 2	1
3	3	1 - 2	2

Sequence Provided By Sample Programming Data:

	Phase 1					Phase 2					Phase 3				
	R	Clear To				R	Clear To				R	Clear To			
Signals	W	2		3		W	3		1		W	1		2	
Phase 1	G	Y	R	Y	R	R	R	R	R	R	R	R	R	R	R
Phase 2	R	R	R	R	R	G	Y	R	Y	R	R	R	R	R	R
Phase 3	R	R	R	R	R	R	R	R	R	R	G	Y	R	Y	R
Ovlp 1 (1+2)	G	G	G	Y	R	G	Y	R	G	G	R	R	R	R	R
Ovlp 2 (1+2)	R	G	G	R	R	G	Y	R	G	G	R	R	R	R	R
Ovlp 3 (1+2)	G	G	G	Y	R	R	R	R	G	G	R	R	R	R	R

NOTE—In the above tables that “RW” means “Right of Way”.

Annex D
DEPRECATED OBJECTS
(Normative)

This Annex provides a repository for deprecated objects.

D.1 SPECIAL FUNCTION OUTPUT STATE

```
specialFunctionOutputState   OBJECT-TYPE
   SYNTAX   INTEGER (0..1)
   ACCESS   read-write
   STATUS   deprecated
   DESCRIPTION
      "<Definition> This object has been replaced by:
          specialFunctionOutputControl and
          specialFunctionOutputStatus.

       The special function output (logical or physical) on
       the device may be controlled by this object.  When
       this object is one then the associated special
       function output signal shall be ON.  When this object
       is zero then the associated special function output
       signal shall be OFF  A read of this object shall
       reflect the current state of the special function
       output.
       <DescriptiveName> NTCIP-1202::ASC.specialFunctionOutputState
       <DataConceptType> Data Element
       <Unit> "
::= { specialFunctionOutputEntry  2 }
```

§