A Project Document of the Joint Committee on the NTCIP

Draft NTCIP 1202 v04 ASC Test Procedures

NTCIP 1202 Version 04
Annex C

National Transportation Communications for ITS Protocol

Object Definitions for Actuated Signal Controllers (ASC) Interface Test Procedures

Draft v04.07b January 28, 2025

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)
1627 Eye Street, N.W., Suite 550
Washington, D.C. 20006

National Electrical Manufacturers Association (NEMA)
1300 North 17th Street, Suite 900
Rosslyn, Virginia 22209-3801

NOTICES

TRF Distribution Permission [when standard contains a TRF]

A Testing Requirements Form ("TRF") may be a Testing Requirements Traceability Table and/or Test Procedures. To the extent that these materials are distributed by AASHTO / ITE / NEMA in the form of a TRF, AASHTO / ITE / NEMA extend the following permission:

a) you may make and/or distribute unlimited electronic or hard copies, including derivative works of the TRF, provided that each copy you make and/or distribute contains the citation "Based on NTCIP 0000 [insert the standard number] TRF. Used by permission. Original text © AASHTO / ITE / NEMA.";
b) you may not modify the logical flow of any test procedure, without clearly noting and marking any such modification; and
c) if the TRF excerpt is made from an unapproved draft, add to the citation "TRF excerpted from a draft standard containing preliminary information that is subject to change."

Test Procedures [Normative]

C.1 Purpose

This annex defines the detailed, but generic, test procedures for testing an implementation of this standard.

C.1.1 Scope

Annex C defines test procedures that cover communication testing for features defined in NTCIP 1202 v04. Test procedures for functional testing may be developed as part of a future revision of NTCIP 1202 v04.

These test procedures are intended to be used as a portion of the overall set of tests that would be performed during component testing of a management station.

Certain test procedures involve testing the Transaction Feature defined in ISO 26048-1. Official documentation for testing global objects, as indicated in the RTM in Annex A.

C.1.2 Keywords

The following key words are used:

- ASSIGN: The test application shall designate a declared variable or specified object as equal to a specific variable or value.
- DELAY: The test application and user shall not perform any actions for a defined period, which may be measured in time units or by monitoring some event that does not involve any exchange of information over the communications media (e.g., DELAY until the temperature exceeds a threshold). In the latter case, the step should also define exception conditions to allow for possibility that the event never happens.
- ERROR INDEX: The value contained in the 'error-index' field of the last SNMPv3Message received from the DUT. See RFC 3416 for additional details related to the error index.
- EXIT: This keyword indicates that the user and test application should terminate the test case without performing any more steps. The keyword by itself does not have any implications as to whether a given test passes or fails.
- FOR: This keyword causes the user (or application) to begin a looping process that shall increment through a series of values. It is comparable to the “for…next” expression in C.
- GET: The test application shall transmit to the DUT one SNMPv3Message containing a GetRequest-PDU, per the rules of NTCIP 2301. Each statement using this keyword shall unambiguously reference the value for the 'name' field(s) to be included in the request. The GetRequest-PDU shall include all of the names in its 'variable-bindings' field. See RFC 3416 for additional details related to the GetRequest-PDU.

  Unless otherwise indicated, the user or test application shall VERIFY the following, in order:

  a) The DUT responds with exactly one SNMPv3Message that contains a Response-PDU, per the rules of NTCIP 2301; this is the RESPONSE. The DUT may also transmit one or more SNMPv3Messages, each of which containing either an SNMPv2-Trap-PDU or an InformRequest-PDU.
  b) The value contained in the 'msgVersion' field of the RESPONSE equals 3 (snmpv3)

c) MESSAGE ID IN equals (MESSAGE ID OUT – 1) [1]
d) MESSAGE MAX SIZE IN is at least 484
e) The value contained in the 'msgFlags' field of the RESPONSE is one octet with the final two (least significant) bits set
f) The value contained in the 'msgSecurityModel' field of the RESPONSE is equal to 4 (TSM)
g) The value contained in the 'msgData' field of the RESPONSE is 'plaintext'
h) CONTEXT ENGINE ID IN equals CONTEXT ENGINE ID OUT
i) CONTEXT NAME IN equals CONTEXT NAME OUT
j) REQUEST ID IN equals (REQUEST ID OUT – 1)[1]
k) RESPONSE ERROR equals 0 (noError)
l) ERROR INDEX equals 0
m) The 'variable-bindings' field contains the same number of VarBind structures as were contained in the GetRequest-PDU
n) The value of each name field in the RESPONSE equals the value of the name field in the GetRequest-PDU that is in the same ordered position.
o) The RESPONSE TYPE of each VarBind Structure in the VarBind List is not equal to noSuchObject

In addition, if MESSAGE MAX SIZE is 0, RECORD MAX MESSAGE SIZE IN as MAX MESSAGE SIZE; otherwise VERIFY that MAX MESSAGE SIZE IN equals MAX MESSAGE SIZE.

- GET-NEXT: The test application shall transmit to the DUT one SNMPv3Message containing a GetNextRequest-PDU, per the rules of NTCIP 2301. Each statement using this keyword shall unambiguously reference the value for the 'name' field(s) to be included in the request. The GetNextRequest-PDU shall include all the names in its 'variable-bindings' field. See RFC 3416 for additional details related to the GetNextRequest-PDU.

  Unless otherwise indicated, the user or test application shall VERIFY the following, in order:

  a) The DUT responds with exactly one SNMPv3Message that contains a Response-PDU, per the rules of NTCIP 2301; this is the RESPONSE. The DUT may also transmit one or more SNMPv3Messages, each of which containing either an SNMPv2-Trap-PDU or an InformRequest-PDU.
  b) The value contained in the 'msgVersion' field of the RESPONSE equals 3 (snmpv3)
  c) MESSAGE ID IN equals (MESSAGE ID OUT – 1)[1]
  d) MESSAGE MAX SIZE IN is at least 484
  e) The value contained in the 'msgFlags' field of the RESPONSE is one octet with the final two (least significant) bits set
  f) The value contained in the 'msgSecurityModel' field of the RESPONSE is equal to 4 (TSM)
  g) The value contained in the 'msgData' field of the RESPONSE is an 'plaintext'
  h) CONTEXT ENGINE ID IN equals CONTEXT ENGINE ID OUT
  i) CONTEXT NAME IN equals CONTEXT NAME OUT
  j) REQUEST ID IN equals (REQUEST ID OUT – 1)[1]
  k) RESPONSE ERROR equals 0 (noError)
  l) ERROR INDEX equals 0
  m) The 'variable-bindings' field contains the same number of VarBind structures as were contained in the GetNextRequest-PDU
  n) The name field in the RESPONSE either 1) contains the same name as in the GetNextRequest-PDU that is in the same ordered position and its value is ‘endOfMibView’ or 2) contains a value that is lexicographically greater than the name in the GetNextRequest-PDU that is in the same ordered position.

[1] The MESSAGE ID OUT and REQUEST ID OUT values logically increment after sending a message; when comparing response messages, the “minus 1” reflects the need to compare the value sent in the request, which is prior to its being incremented.

In addition, if MESSAGE MAX SIZE is 0, RECORD MAX MESSAGE SIZE IN as MAX MESSAGE SIZE; otherwise VERIFY that MAX MESSAGE SIZE IN equals MAX MESSAGE SIZE.

- GOTO: This keyword shall cause the user (or application) to immediately jump to the indicated location in the test procedure (e.g., to another Step).
- IF: This keyword shall cause the user (or application) to perform a comparison and take one action if the comparison evaluates to true and another action if the comparison evaluates to false. It is comparable to the “if…else…” expression in C.
- PERFORM: The user or test application shall perform another test case as a part of this test case. Unless otherwise indicated in the "PERFORM" statement, the user (and test application) shall use the variable values defined when the other test case is performed in a stand-alone fashion.
- PRE-CONDITION: The PRE-CONDITION keyword shall be used as a predicate to the text of a test step to indicate that the text provides a textual description of any pre-conditions for the test case. Pre-conditions are conditions that must be met prior to running a test case. Only one pre-condition shall exist in a test case and it shall always be the first step listed, if present.
- RECORD: The user (or test application) shall record the information indicated by the test step as a part of the test results. This information may be referenced by a later step of the test case (or by a later step of a calling step case).
- RESPONSE ERROR: The value contained in the 'error-status' field of the last SNMPv3Message received from the DUT. See RFC 3416 for additional details related to the error status.
- SET: The test application shall transmit to the DUT one SNMPv3Message containing a SetRequest-PDU, per the rules of NTCIP 2301. Each statement using this keyword shall unambiguously reference the order and value for the 'name' field(s) to be included in the request. The statement shall also indicate the value of the 'value' field associated with each 'name' field. Unless otherwise indicated, the value will be encoded according to the SYNTAX of the associated object. The SetRequest-PDU shall include all of the names and values, with their indicated associations in its 'variable-bindings' field. See RFC 3412 for additional details related to the SetRequest-PDU.

Unless otherwise indicated, the user or test application shall VERIFY that:

a) The DUT responds with exactly one SNMP Message that contains a GetResponse-PDU, per the rules of NTCIP 2301; this is the RESPONSE. The DUT may also respond with one or more SNMPv3Messages, each of which containing either an SNMPv2-Trap-PDU or an InformRequest-PDU
b) The value contained in the 'msgVersion' field of the RESPONSE equals 3 (snmpv3)
c) MESSAGE ID IN equals (MESSAGE ID OUT – 1)[1]
d) MESSAGE MAX SIZE IN is at least 484
e) The vaue contained in the 'msgFlags' field of the RESPONSE is one octet with the final two (least significant) bits set
f) The value contained in the 'msgSecurityModel' field of the RESPONSE is equal to 4 (TSM)
g) The value contained in the 'msgData' field of the RESPONSE is an 'encryptedPDU'
h) CONTEXT ENGINE ID IN equals CONTEXT ENGINE ID OUT
i) CONTEXT NAME IN equals CONTEXT NAME OUT
j) REQUEST ID IN equals (REQUEST ID OUT – 1)[1]
k) RESPONSE ERROR equals 0 (noError)
l) ERROR INDEX equals 0
m) The 'variable-bindings' field contains the same number of VarBind structures as contained in the SetRequest-PDU
n) The value of each name field in the RESPONSE equals the value of the name field in the SetRequest-PDU that is in the same ordered position.

o) The value of each value field in the RESPONSE equals the value of the value field in the SetRequest-PDU that is in the same ordered position.

In addition, if MESSAGE MAX SIZE is 0, RECORD MAX MESSAGE SIZE IN as MAX MESSAGE SIZE; otherwise VERIFY that MAX MESSAGE SIZE IN equals MAX MESSAGE SIZE.

- VERIFY: The user or test application shall evaluate the expression that follows this keyword. Each statement using this keyword shall contain an unambiguous expression that will always evaluate to either true or false without subjective or qualitative judgments by the tester.

If the result is true the following will take place:

a) The verification step shall pass, and
b) The test shall continue to the next step, unless otherwise indicated in the test case.

Otherwise, if the result is false the following will take place:

a) The verification step shall fail.
b) The test case shall fail.
c) The test case shall EXIT, unless otherwise indicated in the test case.

NOTE—While criteria are often stated in exact terms (e.g., "The response shall be ‘3’" or, "The sign shall display 'TEST,'" etc.), it may also be the case that criteria may be stated as ranges or thresholds (e.g., "The response shall be between ‘2’ and ‘16’ inclusive;" or, "The response shall be '3' or greater;" etc.). Each approach is valid and should be considered in the construction of a test case.

C.1.3 Rules for Executing Test Procedures

The test procedures contained in this annex are designed to be used for component testing a device for conformance to the NTCIP 1204 v04 interface standard. To component test a device for conformance to the NTCIP 1204 v04 interface standard, the user shall follow the steps as written, filling in the pass/fail information in the ‘Results' column.

A given test procedure may entail multiple steps that may require multiple interactions between the user and the management station to fulfill the complete test procedure. For example, a single test procedure may transfer the definition of a message to the device and then retrieve the contents of the message to ensure that the values were updated; this might require two user interface operations.

C.2 Testing Requirements

C.2.1 Field Device Test Environment

All Test Cases covered by this Testing Requirements documentation require the Device Under Test (DUT) to be connected to a test application as depicted in Figure C-1. A data analyzer may also be used to capture the data exchanged between the two components. The test environment should be designed to minimize any complicating factors that may result in anomalies unrelated to the specific test case. Failure to isolate such variables in the test environment may result in false results to the test. For example, the device may be conformant with the standard, but communication delays could result in timeouts and be misinterpreted as failures.

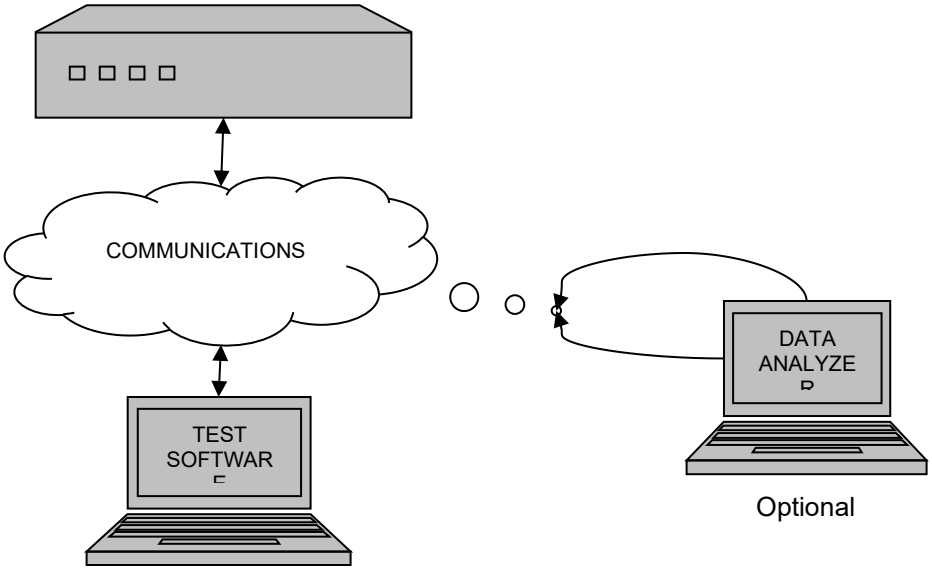

Figure C-1: Field Device Test Environment

The following pre-conditions apply to all test cases unless otherwise defined:

a) All components should be turned on and be provided sufficient time to start up prior to starting any test case
b) The test software should be connected to the central port of the DUT and the DUT should be set for central control
c) The test software, data analyzer, and DUT should all be configured to use a common set of communication settings, including data rates, lower layer protocols, community names, etc.
d) The DUT should be exposed to a medium amount of ambient light so that tests can increase or decrease the amount of light as needed
e) The DUT should have definitions for all font sets that it supports
f) The DUT should have a valid illumination brightness curve defined with a positive slope.

C.2.2 Requirements to Test Case Traceability Table (RTCTT) Table

The Requirements to Test Case Traceability Table defines the traceability between the requirements in Section 3 and the Test Cases presented in this Annex. This table defines the minimal test procedure(s) that shall be completed to confirm that an implementation fulfills a requirement and still conform to this standard.

To confirm that an implementation fulfills a requirement, the DUT shall successfully pass all test cases that trace to that requirement.

Requirement ID	Functional Requirement	Test Case ID	Test Case Title
3.4	Architectural Requirements		
3.4.1	Support Basic Communications Requirements		
3.4.1.1	Retrieve Data		
		C.3.3.51	Determine Maximum Number of Phases

3.4.1.2	Deliver Data		
		C.3.2.2	Configure Startup Flash Time
3.4.1.3	Explore Data		
		C.3.1.3	Explore Data
3.4.1.4	Monitor SNMP Requirements		
3.4.1.4.1	Monitor SNMP Information		
		C.3.1.4	Monitor SNMP Information
3.4.2	Manage Data Blocks Requirements		
3.4.2.1	Store Pre-Defined Compressed Data Blocks		
3.4.3	Support Logged Data Requirements		
3.4.4	Support Database Management		
3.4.5	Support Condition-based Exception Reporting Requirements		
3.5	Data Exchange and Operational Environment Requirements		
3.5.1	ASC Configuration Management Requirements		
3.5.1.1	Manage ASC Information Requirements		
3.5.1.1.1	Configure ASC Location - Antenna Offset		
		C.3.2.4	Configure ASC Antenna Offset
3.5.1.2	Manage Communications Requirements		
3.5.1.2.1	Configure Communications Requirements		
3.5.1.2.1.1	Enable/Disable Communications Port		N/A
3.5.1.2.2	Retrieve Communications Requirements		
3.5.1.2.2.1	Determine Number of ASC Communications Ports		
		C.3.9.1	Determine Number of ASC Communications Ports
3.5.1.3	Manage Cabinet Environment Requirements		
3.5.1.4	Monitor Power Source Requirements		

3.5.1.5	Manage Operational Performance Data Requirements		
3.5.1.6	Manage Auxiliary External Inputs/Outputs Requirements		
3.5.1.7	Manage Database Operations		
3.5.1.7.1	Determine Configuration Identifier Parameter Content		
3.5.1.8	Manage Interface with External Detectors Requirements		
3.5.1.9	Manage ASC Clock Requirements		
3.5.1.10	Manage External Control Local Application State Requirements		
3.5.1.10.1	Manage ECLA Interface Requirements		
3.5.1.10.1.1	Enable ECLA Communications		
		C.3.9.18	Enable/Disable ECLA Communications
3.5.1.10.1.2	Disable ECLA Communications		
		C.3.9.18	Enable/Disable ECLA Communications
		C.3.12.1	Enable/Disable Manual Backup
3.5.1.10.2	Monitor ECLA Interface Data Input Time		N/A
3.5.2	Manage Signal Operations Management Requirements		
3.5.2.1	Manage Signal Configuration Requirements		
3.5.2.1.1	Manage Unit Configuration Requirements		
3.5.2.1.1.1	Manage Start-Up Flash Requirements		
3.5.2.1.1.1.1	Configure Start-Up Flash Mode		
		C.3.2.1	Configure Startup All-Red Flash Mode
3.5.2.1.1.1.2	Configure Start-Up Flash Time		
		C.3.2.2	Configure Startup Flash Time
3.5.2.1.1.2	Configure Backup Time		
		C.3.2.3	Configure Backup Time
3.5.2.1.2	Manage Phase Configuration Requirements		
3.5.2.1.2.1	Configure Phases Requirements		

3.5.2.1.2.1.1	Enable/Disable Phase		
		C.3.3.1	Enable/Disable Phase
3.5.2.1.2.1.2	Configure Phase Minimum Green Time		
		C.3.3.2	Configure Phase Minimum Green Time
3.5.2.1.2.1.3	Configure Phase Passage Time		
		C.3.3.3	Configure Phase Passage Time
3.5.2.1.2.1.4	Configure Two Phase Maximum Green Times		
		C.3.3.4	Configure Two Phase Maximum Green Times
3.5.2.1.2.1.5	Configure Three Phase Maximum Green Times		
		C.3.3.4	Configure Two Phase Maximum Green Times
		C.3.3.5	Configure Three Phase Maximum Green Times
3.5.2.1.2.1.6	Configure Phase Yellow Change Time		
		C.3.3.6	Configure Phase Yellow Change Time
3.5.2.1.2.1.7	Configure Phase Red Clearance Time		
		C.3.3.7	Configure Phase Red Clearance Time
3.5.2.1.2.1.8	Configure Phase Red Revert Time		
		C.3.3.8	Configure Phase Red Revert Time
3.5.2.1.2.1.9	Configure Unit Red Revert Time		
		C.3.3.9	Configure Unit Red Revert Time
3.5.2.1.2.1.10	Configure Phase Added Initial Time		
		C.3.3.10	Configure Phase Added Initial Time
3.5.2.1.2.1.11	Configure Phase Maximum Initial Time		
		C.3.3.11	Configure Phase Maximum Initial Time
3.5.2.1.2.1.12	Configure Phase Time Before Reduction		
		C.3.3.12	Configure Phase Time Before Reduction
3.5.2.1.2.1.13	Configure Phase Time to Reduce		
		C.3.3.13	Configure Phase Time to Reduce
3.5.2.1.2.1.14	Configure Phase Cars Before Reduction		
		C 3.3.14	Configure Cars Before Reduction

3.5.2.1.2.1.15	Configure Phase Reduce By Time		
		C.3.3.15	Configure Phase Reduce By Time
3.5.2.1.2.1.16	Configure Phase Minimum Gap Time		
		C.3.3.16	Configure Phase Minimum Gap Time
3.5.2.1.2.1.17	Configure Phase Dynamic Maximum Limit		
		C.3.3.17	Configure Phase Dynamic Maximum Limit
3.5.2.1.2.1.18	Configure Phase Dynamic Maximum Step		
		C.3.3.18	Configure Phase Dynamic Maximum Step
3.5.2.1.2.1.19	Configure Phase Startup State		
		C.3.3.19	Configure Phase Startup State
3.5.2.1.2.1.20	Configure Automatic Flash Entry Phase		
		C.3.3.20	Configure Automatic Flash Entry Phase
3.5.2.1.2.1.21	Configure Automatic Flash Exit Phase		
		C.3.3.21	Configure Automatic Flash Exit Phase
3.5.2.1.2.1.22	Configure Call to Non-Actuated 1		
		C.3.3.22	Configure Call to Non-Actuated 1
3.5.2.1.2.1.23	Configure Call to Non-Actuated 2		
		C.3.3.23	Configure Call to Non-Actuated 2
3.5.2.1.2.1.24	Configure Non-Lock Detector Memory		
		C.3.3.24	Configure Non-Lock Detector Memory
3.5.2.1.2.1.25	Configure Phase Minimum Vehicle Recall		
		C.3.3.25	Configure Minimum Recall
3.5.2.1.2.1.26	Configure Phase Maximum Vehicle Recall		
		C.3.3.26	Configure Maximum Vehicle Recall
3.5.2.1.2.1.27	Configure Phase Soft Vehicle Recall		
		C.3.3.27	Configure Soft Recall
3.5.2.1.2.1.28	Configure Dual Phase Entry		
		C.3.3.28	Configure Dual Phase Entry
3.5.2.1.2.1.29	Configure Simultaneous Gap Disable		
		C.3.3.29	Configure Simultaneous Gap Disable

3.5.2.1.2.1.30	Configure Guaranteed Passage		
		C.3.3.30	Configure Guaranteed Passage
3.5.2.1.2.1.31	Configure Actuated Rest-in-Walk		
		C.3.3.31	Configure Actuated Rest-in-Walk
3.5.2.1.2.1.32	Configure Conditional Service Enable		
		C.3.3.32	Configure Conditional Service Enable
3.5.2.1.2.1.33	Configure Added Initial Calculation		
		C.3.3.33	Configure Added Initial Calculation
3.5.2.1.2.1.34	Configure Phase-to-Ring Association		
		C.3.3.34	Configure Phase-to-Ring Association
3.5.2.1.2.1.35	Configure Phase Concurrency		
		C.3.3.35	Configure Phase Concurrency
3.5.2.1.2.1.36	Configure Pedestrian Clearance Time Allowed During Vehicle Clearance		
		C.3.3.36	Configure Pedestrian Clearance Time Allowed During Vehicle Clearance
3.5.2.1.2.1.37	Configure Pedestrian Walk Time		
		C.3.3.37	Configure Pedestrian Walk Time
3.5.2.1.2.1.38	Configure Pedestrian Clearance Time		
		C.3.3.38	Configure Pedestrian Clearance Time
3.5.2.1.2.1.39	Configure Pedestrian Phase Walk Service Limit		
		C.3.3.39	Configure Pedestrian Clearance Time
3.5.2.1.2.1.40	Configure Pedestrian Phase Don't Walk Revert Time		
		C.3.3.40	Configure Pedestrian Phase Don't Walk Revert Time
3.5.2.1.2.1.41	Configure Non-Lock Ped Detector Memory		
		C.3.3.24	Configure Non-Lock Detector Memory
3.5.2.1.2.1.42	Configure Pedestrian Phase Recall		
		C.3.3.41	Configure Pedestrian Phase Recall
3.5.2.1.2.1.43	Configure Phase Alternate Pedestrian Clearance Time		
		C.3.3.42	Configure Phase Alternate Pedestrian Clearance Time

3.5.2.1.2.1.44	Configure Alternate Pedestrian Walk Time		
		C.3.3.43	Configure Phase Alternate Pedestrian Walk Time
3.5.2.1.2.1.45	Configure Pedestrian Phase Advanced Walk Time		
		C.3.3.44	Configure Pedestrian Phase Advanced Walk Time
3.5.2.1.2.1.46	Configure Pedestrian Phase Delayed Walk Time		
		C.3.3.45	Configure Pedestrian Phase Delayed Walk Time
3.5.2.1.2.1.47	Configure Phase Advance Warning Green		
		C.3.3.46	Configure Phase Advance Warning Green
3.5.2.1.2.1.48	Configure Phase Advance Warning Red		
		C.3.3.47	Configure Phase Advance Warning Red
3.5.2.1.2.1.49	Configure Flashing Yellow Arrow Associated Vehicle Phase		
		C.3.7.11	Configure Overlap Included Phases
		C.3.7.12	Configure Overlap Modifier Phases
3.5.2.1.2.1.50	Configure Flashing Red Arrow Associated Vehicle Phase		
		C.3.7.11	Configure Overlap Included Phases
		C.3.7.12	Configure Overlap Modifier Phases
3.5.2.1.2.1.51	Configure Alternate Minimum Vehicle Green Time during Transition		
		C.3.3.48	Configure Alternate Minimum Vehicle Green Time during Transition
3.5.2.1.2.1.52	Configure Alternate Minimum Pedestrian Walk Time during Transition		
		C.3.3.49	Configure Alternate Minimum Pedestrian Walk Time during Transition
3.5.2.1.2.1.53	Configure Alternate Minimum Pedestrian Clearance Time during Transition		
		C.3.3.50	Configure Alternate Minimum Pedestrian Clearance Time during Transition
3.5.2.1.2.2	Configure Multiple Phase Sets		N/A
3.5.2.1.2.3	Retrieve Phase Configuration Requirements		

3.5.2.1.2.3.1	Determine Maximum Number of Phases		
		C.3.3.51	Determine Maximum Number of Phases
3.5.2.1.2.3.2	Determine Maximum Number of Phase Sets		
		C.3.3.52	Determine Maximum Number of Phase Sets
3.5.2.1.3	Manage Coordination Configuration Requirements		
3.5.2.1.3.1	Configuration Operational Mode for Coordination		
		C.3.4.1	Configure Operational Mode for Coordination - Automatic
3.5.2.1.3.2	Configuration Correction Mode for Coordination		
		C.3.4.2	Configure Correction Mode for Coordination
3.5.2.1.3.3	Configure Maximum Mode for Coordination		
		C.3.4.3	Configure Correction Mode for Coordination
3.5.2.1.3.4	Configure Unit-Level Force Mode for Coordination		
		C.3.4.4	Configure Unit-level Force Mode for Coordination
3.5.2.1.3.5	Configure Phase-Level Force Mode for Coordination		
		C.3.4.5	Configure Phase-Level Force Mode for Coordination
3.5.2.1.3.6	Configure Pattern Reference Phase		
		C.3.4.6	Configure Pattern Reference Phase
3.5.2.1.3.7	Configure Pattern Reference Point		
		C.3.4.7	Configure Pattern Reference Point
3.5.2.1.3.8	Configure Omit Phases During Transitions		
		C.3.4.8	Configure Omit Phases During Transitions
3.5.2.1.3.9	Configure Pattern Synchronization Time		
		C.3.4.9	Configure Pattern Synchronization Time
3.5.2.1.4	Manage Timing Patterns Requirements		
3.5.2.1.4.1	Configure Timing Patterns Requirements		
3.5.2.1.4.1.1	Configure Pattern to Run Free		
		C.3.4.10	Configure Pattern to Run Free

3.5.2.1.4.1.2	Configure Pattern to Run Automatic Flash		
		C.3.4.11	Configure Pattern to Run Automatic Flash
3.5.2.1.4.1.3	Configure Pattern Cycle Time for Coordination		
		C.3.4.12	Configure Pattern Cycle Time for Coordination
3.5.2.1.4.1.4	Configure Pattern Offset Time		
		C.3.4.13	Configure Pattern Offset Time
3.5.2.1.4.1.5	Configure Pattern Split Association		
		C.3.4.14	Configure Pattern Split Association
3.5.2.1.4.1.6	Configure Pattern Sequence Association		
		C.3.4.15	Configure Pattern Sequence Association
3.5.2.1.4.1.7	Configure Pattern Maximum Mode		
		C.3.4.16	Configure Pattern Maximum Mode
3.5.2.1.4.1.8	Configure Pattern Phase Set		
		C.3.4.17	Configure Pattern Phase Set
3.5.2.1.4.1.9	Configure Pattern Overlap Set		
		C.3.4.18	Configure Pattern Overlap Set
3.5.2.1.4.1.10	Configure Pattern Vehicle Detector Set		
		C.3.4.19	Configure Pattern Vehicle Detector Set
3.5.2.1.4.1.11	Configure Pattern Pedestrian Detector Set		
		C.3.4.20	Configure Pattern Pedestrian Detector Set
3.5.2.1.4.1.12	Configure Pattern Special Functions		
		C.3.4.21	Configure Pattern Special Functions
3.5.2.1.4.1.13	Determine Maximum Number of Timing Patterns		
		C.3.4.22	Determine Maximum Number of Timing Patterns
3.5.2.1.5	Manage Splits Configuration Requirements		
3.5.2.1.5.1	Configure Split Requirements		
3.5.2.1.5.1.1	Configure Phase Split Time		
		C.3.4.23	Configure Phase Split Time
3.5.2.1.5.1.2	Configure Phase Split Mode		
		C.3.4.24	Configure Phase Split Mode

3.5.2.1.5.1.3	Configure Split Coordination Phase		
		C.3.4.25	Configure Split Coordinated Phase
3.5.2.1.5.2	Determine Maximum Number of Phase Splits		
		C.3.4.26	Determine Maximum Number of Phase Splits
3.5.2.1.6	Manage Ring Configuration Requirements		
3.5.2.1.6.1	Configure Sequence Data		
		C.3.5.1	Configure Sequence Data
3.5.2.1.6.2	Determine Maximum Number of Rings		
		C.3.5.2	Determine Maximum Number of Rings
3.5.2.1.6.3	Determine Maximum Number of Sequences		
		C.3.5.2	Determine Maximum Number of Sequences
3.5.2.1.7	Manage Channel Configuration Requirements		
3.5.2.1.7.1	Configure Channel Requirements		
3.5.2.1.7.1.1	Configure Channel Control Source		
		C.3.6.1	Configure Channel Control Source
3.5.2.1.7.1.2	Configure Channel Control Type		
		C.3.6.2	Configure Channel Control Type
3.5.2.1.7.1.3	Configure Channel Flash Requirements		
3.5.2.1.7.1.3.1	Configure Channel Flash Yellow		
		C.3.6.3	Configure Channel Flash Yellow
3.5.2.1.7.1.3.2	Configure Channel Flash Red		
		C.3.6.4	Configure Channel Flash Red
3.5.2.1.7.1.3.3	Configure Channel Flash Alternate Half Hertz		
		C.3.6.5	Configure Channel Flash Alternate Half Hertz
3.5.2.1.7.1.3.4	Configure Channel Flash Alternate First or Second		
		C.3.6.6	Configure Channel Flash Alternate First or Second
3.5.2.1.7.2	Determine Maximum Number of Channels		
		C.3.6.7	Determine Maximum Number of Channels Supported

3.5.2.1.8	Manage Overlap Configuration Requirements		
3.5.2.1.8.1	Configure Overlap Requirements		
3.5.2.1.8.1.1	Configure Overlap Type Requirements		
3.5.2.1.8.1.1.1	Configure Overlap Type - Vehicle Normal		
		C.3.7.1	Configure Overlap Type - Normal
3.5.2.1.8.1.1.2	Configure Overlap Type - Vehicle Minus Green and Yellow		
		C.3.7.2	Configure Overlap Type - Vehicle Minus Green and Yellow
3.5.2.1.8.1.1.3	Configure Overlap Type - Pedestrian Normal		
		C.3.7.3	Configure Overlap Type - Pedestrian Normal
3.5.2.1.8.1.1.4	Configure Overlap Type - Flashing Yellow Arrow - 3 Section Head		
		C.3.7.4	Configure Overlap Type - Flashing Yellow Arrow - 3 Section Head
3.5.2.1.8.1.1.5	Configure Overlap Type - Flashing Yellow Arrow - 4 Section Head		
		C.3.7.5	Configure Overlap Type - Flashing Yellow Arrow - 4 Section Head
3.5.2.1.8.1.1.6	Configure Overlap Type - Flashing Red Arrow - 3 Section Head		
		C.3.7.6	Configure Overlap Type - Flashing Red Arrow - 3 Section Head
3.5.2.1.8.1.1.7	Configure Overlap Type - Flashing Red Arrow - 4 Section Head		
		C.3.7.7	Configure Overlap Type - Flashing Red Arrow - 4 Section Head
3.5.2.1.8.1.1.8	Configure Overlap Type - 2 Section Transit Specific Signal Head		
		C.3.7.8	Configure Overlap Type - 2 Section Transit Specific Signal Head
3.5.2.1.8.1.1.9	Configure Overlap Type - Minus Green Yellow Alternate		
		C.3.7.9	Configure Overlap Type - Minus Green Yellow Alternate

3.5.2.1.8.1.2	Configure Overlap Included Phases		
		C.3.7.10	Configure Overlap Included Phases
3.5.2.1.8.1.3	Configure Overlap Modifier Phases		
		C.3.7.11	Configure Overlap Modifier Phases
3.5.2.1.8.1.4	Configure Pedestrian Modifier Phases		
		C.3.7.12	Configure Pedestrian Modifier Phases
3.5.2.1.8.1.5	Configure Overlap Trailing Green		
		C.3.7.13	Configure Overlap Trailing Green
3.5.2.1.8.1.6	Configure Overlap Trailing Yellow		
		C.3.7.14	Configure Overlap Trailing Yellow
3.5.2.1.8.1.7	Configure Overlap Trailing Red Clearance		
		C.3.7.15	Configure Overlap Trailing Red Clearance
3.5.2.1.8.1.8	Configure Overlap Walk		
		C.3.7.16	Configure Overlap Walk
3.5.2.1.8.1.9	Configure Overlap Pedestrian Clearance		
		C.3.7.17	Configure Overlap Pedestrian Clearance
3.5.2.1.8.2	Configure Multiple Overlap Sets		N/A
3.5.2.1.8.3	Retrieve Overlaps Requirements		
3.5.2.1.8.3.1	Determine Maximum Number of Overlaps		
		C.3.7.20	Determine Maximum Number of Overlaps
3.5.2.1.8.3.2	Determine Maximum Number of Overlap Sets		
		C.3.7.19	Determine Maximum Number of Overlap Sets
3.5.2.1.9	Manage Preempt Configuration Requirements		
3.5.2.1.9.1	Configure Preempts Requirements		
3.5.2.1.9.1.1	Enable/Disable Preempt Inputs		
		C.3.8.1	Enable/Disable Preempt Inputs
3.5.2.1.9.1.2	Configure Preempt Control - Non-Locking Memory		
		C.3.8.2	Configure Preempt Control - Non-Locking Memory

3.5.2.1.9.1.3	Configure Preempt Control - Override Automatic Flash		
		C.3.8.3	Configure Preempt Control - Preempt Override Flash
3.5.2.1.9.1.4	Configure Preempt Control - Override Preempt		
		C.3.8.4	Configure Preempt Control - Preempt Override Priority
3.5.2.1.9.1.5	Configure Preempt Control - Flash Dwell		
		C.3.8.5	Configure Preempt Control - Flash Dwell
3.5.2.1.9.1.6	Configure Preempt Control - All Red Entry		
		C.3.8.6	Configure Preempt Control – All Red Entry
3.5.2.1.9.1.7	Configure Preempt Link		
		C.3.8.7	Configure Preempt Link
3.5.2.1.9.1.8	Configure Preempt Delay		
		C.3.8.8	Configure Preempt Delay
3.5.2.1.9.1.9	Configure Preempt Minimum Duration		
		C.3.8.9	Configure Preempt Minimum Duration
3.5.2.1.9.1.10	Preempt Entry Configuration Requirements		
3.5.2.1.9.1.10.1	Configure Preempt Enter Minimum Green Time		
		C.3.8.10	Configure Preempt Minimum Green Time
3.5.2.1.9.1.10.2	Configure Preempt Enter Minimum Walk Time		
		C.3.8.11	Configure Preempt Minimum Walk Time
3.5.2.1.9.1.10.3	Configure Preempt Enter Pedestrian Clearance Time		
		C.3.8.12	Configure Preempt Enter Pedestrian Clearance Time
3.5.2.1.9.1.10.4	Configure Preempt Enter Yellow Change Time		
		C.3.8.13	Configure Preempt Enter Yellow Change Time
3.5.2.1.9.1.10.5	Configure Preempt Enter Red Clearance Time		
		C.3.8.14	Configure Preempt Enter Red Clearance Time
3.5.2.1.9.1.11	Configure Preempt Track Clearance Requirements		
3.5.2.1.9.1.11.1	Configure Preempt Track Clearance Green Time		

		C.3.8.15	Configure Preempt Track Clearance Green Time
3.5.2.1.9.1.11.2	Configure Preempt Track Clearance Yellow Change Time		
		C.3.8.16	Configure Preempt Track Clearance Yellow Change Time
3.5.2.1.9.1.11.3	Configure Preempt Track Red Clearance Time		
		C.3.8.17	Configure Preempt Track Red Clearance Time
3.5.2.1.9.1.11.4	Configure Preempt Track Clearance Phases		
		C.3.8.18	Configure Preempt Track Clearance Phases
3.5.2.1.9.1.11.5	Configure Preempt Track Clearance Overlaps		
		C.3.8.19	Configure Preempt Track Clearance Overlaps
3.5.2.1.9.1.12	Configure Preempt Dwell Requirements		
3.5.2.1.9.1.12.1	Configure Preempt Minimum Green Dwell Time		
		C.3.8.20	Configure Preempt Minimum Dwell Time
3.5.2.1.9.1.12.2	Configure Preempt Dwell Phases		
		C.3.8.21	Configure Preempt Dwell Phases
3.5.2.1.9.1.12.3	Configure Preempt Dwell Pedestrian Movements		
		C.3.8.22	Configure Preempt Dwell Pedestrian Movements
3.5.2.1.9.1.12.4	Configure Preempt Dwell Overlaps		
		C.3.8.23	Configure Preempt Dwell Overlaps
3.5.2.1.9.1.12.5	Configure Preempt Cycling Phases		
		C.3.8.24	Configure Preempt Cycling Phases
3.5.2.1.9.1.12.6	Configure Preempt Cycling Pedestrian Movements		
		C.3.8.25	Configure Preempt Cycling Pedestrian Movements
3.5.2.1.9.1.12.7	Configure Preempt Cycling Phases Sequence		
		C.3.8.26	Configure Preempt Cycling Phases Sequence
3.5.2.1.9.1.12.8	Configure Preempt Cycling Overlaps		
		C.3.8.27	Configure Preempt Cycling Overlaps

3.5.2.1.9.1.13	Configure Preempt Exit Requirements		
3.5.2.1.9.1.13.1	Configure Preempt Exit Phases		
		C.3.8.28	Configure Preempt Exit Phases
3.5.2.1.9.1.13.2	Configure Preempt Exit Phase Strategy		
		C.3.8.29	Configure Preempt Exit Phase Strategy
3.5.2.1.9.1.13.3	Configure Preempt Exit Priority Levels		
		C.3.8.30	Configure Preempt Exit Priority Levels
3.5.2.1.9.1.14	Configure Preempt Max Presence Exceeded Requirements		
3.5.2.1.9.1.14.1	Configure Preempt Maximum Presence Time		
		C.3.8.31	Configure Preempt Maximum Presence Time
3.5.2.1.9.1.14.2	Configure Preempt Maximum Presence Action		
		C.3.8.32	Configure Preempt Maximum Presence Action
3.5.2.1.9.1.15	Configure Preempt Gate Description		
		C.3.8.33	Configure Preempt Gate Description
3.5.2.1.9.2	Determine Maximum Number of Preempts		
		C.3.8.34	Determine Maximum Number of Preempts
3.5.2.1.10	Manage ASC Scheduler Requirements		
3.5.2.1.10.1	Configure ASC Timebased Action Requirements		
3.5.2.1.10.1.1	Configure Timebased Action - Pattern		
		C.3.4.27	Configure Timebased Action - Pattern
3.5.2.1.10.1.2	Configure Timebased Action - Special Functions		
		C.3.4.28	Configure Timebased Action – Special Functions
3.5.2.1.10.1.3	Determine Maximum Number of Timebased Actions		
		C.3.4.29	Determine Maximum Number of Timebased Actions
3.5.2.1.10.1.4	Determine Action In Effect		
		C.3.4.30	Determine Action In Effect

3.5.2.1.10.1.5	Activate Action Plan Remotely		
		C.3.4.31	Activate Action Plan Remotely
3.5.2.1.11	Manage I/O Mapping Requirements		
3.5.2.1.11.1	Configure I/O Mapping Requirements		
3.5.2.1.11.1.1	Set Active I/O Map		N/A
3.5.2.1.11.1.2	Configure I/O Map Requirements		
3.5.2.1.11.1.2.1	Configure I/O Map Description		N/A
3.5.2.1.11.1.2.2	Configure I/O Map Input Requirements		
3.5.2.1.11.1.2.2.1	Configure I/O Map Input Device		N/A
3.5.2.1.11.1.2.2.2	Configure I/O Map Input Device Pin		N/A
3.5.2.1.11.1.2.2.3	Configure I/O Map Input Function		N/A
3.5.2.1.11.1.2.3	Configure I/O Map Output Requirements		
3.5.2.1.11.1.2.3.1	Configure I/O Map Output Device		N/A
3.5.2.1.11.1.2.3.2	Configure I/O Map Output Device Pin		N/A
3.5.2.1.11.1.2.3.3	Configure I/O Map Output Function		N/A
3.5.2.1.11.2	Determine I/O Mapping Requirements		
3.5.2.1.11.2.1	Retrieve Maximum Number of I/O Maps		
		C.3.9.2	Retrieve Maximum Number of I/O Maps
3.5.2.1.11.2.2	Retrieve Maximum Number of I/O Map Inputs		
		C.3.9.3	Retrieve Maximum Number of I/O Map Inputs
3.5.2.1.11.2.3	Retrieve Maximum Number of I/O Map Outputs		
		C.3.9.4	Retrieve Maximum Number of I/O Map Outputs
3.5.2.1.11.2.4	Retrieve I/O Mapping Activate Conditions		
		C.3.9.5	Retrieve I/O Mapping Activate Conditions
3.5.2.1.11.2.5	Retrieve I/O Mapping Input Functions		
		C.3.9.6	Retrieve I/O Mapping Input Functions

3.5.2.1.11.2.6	Retrieve I/O Mapping Output Functions		
		C.3.9.7	Retrieve I/O Mapping Output Functions
3.5.2.1.11.2.7	Retrieve I/O Map Input Device Pin Status		
		C.3.9.8	Retrieve I/O Map Input Device Pin Status
3.5.2.1.11.2.8	Retrieve I/O Map Output Device Pin Status		
		C.3.9.9	Retrieve I/O Map Output Device Pin Status
3.5.2.1.11.2.9	Enumerate I/O Mapping Device Pin Requirements		
3.5.2.1.11.2.9.1	Enumerate I/O Map - FIO Inputs		N/A
3.5.2.1.11.2.9.2	Enumerate I/O Map - FIO Outputs		N/A
3.5.2.1.11.2.9.3	Enumerate I/O Map - TS1 Inputs		N/A
3.5.2.1.11.2.9.4	Enumerate I/O Map - TS1 Outputs		N/A
3.5.2.1.11.2.9.5	Enumerate I/O Map - TS2 BIU Inputs		N/A
3.5.2.1.11.2.9.6	Enumerate I/O Map - TS2 BIU Outputs		N/A
3.5.2.1.11.2.9.7	Enumerate I/O Map - ATC Cabinet SIU Inputs		N/A
3.5.2.1.11.2.9.8	Enumerate I/O Map - ATC Cabinet SIU Outputs		N/A
3.5.2.1.11.2.9.9	Enumerate I/O Map - Auxiliary Device Inputs		N/A
3.5.2.1.11.2.9.10	Enumerate I/O Map - Auxiliary Device Outputs		N/A
3.5.2.1.12	Manage Intra-Cabinet Communications Requirements		
3.5.2.1.12.1	Manage Intra-Cabinet Communications Requirements - ATC		
3.5.2.1.12.1.1	Determine Serial Bus 1 Device Present		
		C.3.9.10	Determine Serial Bus 1 Device Present
3.5.2.1.12.2	Manage Intra-Cabinet Communications Requirements - TS2		
3.5.2.1.12.2.1	Determine TS2 Port 1 Device Present		
		C.3.9.11	Determine TS2 Port 1 Device Present

3.5.2.1.12.2.2	Enable/Disable TS2 Port 1 Frame 40 Messages		
		C.3.9.12	Determine TS2 Port 1 Frame 40 Enable
3.5.2.1.13	Manage ADA Support Requirements		
3.5.2.1.13.1	Configure ADA Support Requirements		
3.5.2.1.13.1.1	Configure APS Push Button Minimum Press Time		
3.5.2.1.13.1.2	Configure APS Push Button to Phase Association		
		C.3.13.19	Configure Pedestrian Detector Call Phase
3.5.2.1.13.1.3	Configure APS Extra Crossing Time		
		C.3.3.42	Configure Alternate Pedestrian Clearance Time
		C.3.3.43	Configure Alternate Pedestrian Walk Time
3.5.2.1.13.1.4	Configure Pedestrian Detector for Alternate Pedestrian Timing		
		C.3.13.24	Configure Pedestrian Detector for Alternate Pedestrian Timing
3.5.2.1.14	Manage Block Object Requirements		
3.5.2.1.14.1	Configure Block Object Requirements		
3.5.2.1.14.1.1	Configure Block Object Get Control - Phase Data		
		C.3.10.1	Configure Phase Data Block
3.5.2.1.14.1.2	Configure Block Object Get Control - Vehicle Detector Data		
		C.3.10.2	Configure Vehicle Detector Block
3.5.2.1.14.1.3	Configure Block Object Get Control - Pedestrian Detector Data		
		C.3.10.3	Configure Pedestrian Detector Block
3.5.2.1.14.1.4	Configure Block Object Get Control - Pattern Data		
		C.3.10.4	Configure Pattern Data Block
3.5.2.1.14.1.5	Configure Block Object Get Control - Split Data		
		C.3.10.5	Configure Split Data Block

3.5.2.1.14.1.6	Configure Block Object Get Control - Overlap Data		
		C.3.10.6	Configure Overlap Data Block
3.5.2.1.14.1.7	Configure Block Object Get Control - Preempt Data		
		C.3.10.7	Configure Preempt Data Block
3.5.2.1.14.1.8	Configure Block Object Get Control - Sequence Data		
		C.3.10.8	Configure Sequence Data Block
3.5.2.1.14.1.9	Configure Block Object Get Control - Channel Data		
		C.3.10.9	Configure Channel Data Block
3.5.2.1.14.2	Monitor Block Error Status - Error-causing Data Element		
		C.3.10.1	Monitor Block Error Status - Value Set Validity Error
3.5.2.2	Monitor Signal Operations Requirements		
3.5.2.2.1	Determine Controller Health Requirements		
3.5.2.2.1.1	Monitor External Alarm Input States		
		C.3.11.1	Monitor External Alarm Input States
3.5.2.2.1.2	Monitor External Alarm Active		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.1.3	Monitor Flash Status		
		C.3.11.2	Monitor Unit Level Alarms - Short
		C.3.11.4	Monitor Flash Status
3.5.2.2.1.4	Monitor Local Override		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.1.5	Monitor Coordination Alarm		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.1.6	Monitor Detector Fault		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.1.7	Monitor Stop Time Active		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.1.8	Monitor Cycle Fault Alarm		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.1.9	Monitor Coordination Fault		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.1.10	Monitor Coordination Fail Alarm		
		C.3.11.3	Monitor Unit Level Alarms

3.5.2.2.1.11	Monitor Cycle Fail Alarm		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.1.12	Monitor Cabinet IO Link Alarm		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.1.13	Monitor SMU Communications Error		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.1.14	Monitor Preempt Maximum Presence Alarm		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.2	Retrieve Current Operation Requirements		
3.5.2.2.2.1	Monitor Unit Control Status		
		C.3.11.5	Monitor Unit Control Status
3.5.2.2.2.2	Monitor Preempt Active		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.2.3	Monitor Offset Transitioning		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.2.4	Monitor Priority Call Active		
		C.3.11.2	Monitor Unit Level Alarms - Short
3.5.2.2.2.5	Monitor Local Free Status		
		C.3.11.3	Monitor Unit Level Alarms
		C.3.11.6	Monitor Local Free Status
3.5.2.2.2.6	Monitor Coordination Active		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.2.7	Monitor ECLA Active		
		C.3.11.3	Monitor Unit Level Alarms
3.5.2.2.2.8	Monitor Current Timing Pattern Requirements		
3.5.2.2.2.8.1	Monitor Current Pattern Status		
		C.3.11.7	Monitor Current Pattern Status
3.5.2.2.2.8.2	Monitor Current Pattern Command Source		
		C.3.11.8	Monitor Current Pattern Command Source
3.5.2.2.2.8.3	Monitor Current Pattern Fault Status		
		C.3.11.9	Monitor Current Pattern Fault Status
3.5.2.2.2.9	Monitor Current Timing Pattern Requirements		
3.5.2.2.2.9.1	Monitor Coordination Cycle Status		

		C.3.11.10	Monitor Coordination Cycle Status
3.5.2.2.2.9.2	Monitor Coordination Synchronization Status		
		C.3.11.11	Monitor Coordination Synchronization Status
3.5.2.2.2.9.3	Monitor Current Offset		
		C.3.11.12	Monitor Current Offset
3.5.2.2.3	Retrieve Current Signal Indication Requirements		
3.5.2.2.3.1	Monitor Active Red Phases		
		C.3.11.13	Monitor Active Red Phases
3.5.2.2.3.2	Monitor Active Yellow Phases		
		C.3.11.14	Monitor Active Yellow Phases
3.5.2.2.3.3	Monitor Active Green Phases		
		C.3.11.15	Monitor Active Green Phases
3.5.2.2.3.4	Monitor Active Don't Walk Phases		
		C.3.11.16	Monitor Active Don't Walk Phases
3.5.2.2.3.5	Monitor Active Pedestrian Clearance Phases		
		C.3.11.17	Monitor Active Pedestrian Clearance Phases
3.5.2.2.3.6	Monitor Active Walk Phases		
		C.3.11.18	Monitor Active Walk Phases
3.5.2.2.3.7	Monitor Active On Phases		
		C.3.11.19	Monitor Active On Phases
3.5.2.2.3.8	Monitor Next Phases		
		C.3.11.20	Monitor Next Phases
3.5.2.2.3.9	Monitor Phase Vehicle Calls		
		C.3.11.21	Monitor Phase Vehicle Calls
3.5.2.2.3.10	Monitor Phase Pedestrian Calls		
		C.3.11.22	Monitor Phase Pedestrian Calls
3.5.2.2.4	Retrieve Current Ring Requirements		
3.5.2.2.4.1	Monitor Ring Status		
		C.3.11.23	Monitor Ring Status
3.5.2.2.4.2	Monitor Ring Termination Cause		
		C.3.11.23	Monitor Ring Status
3.5.2.2.4.3	Monitor Current Phase On Time		
		C.3.11.24	Monitor Current Phase On Time

3.5.2.2.5	Retrieve Current Channel Status Requirements		
3.5.2.2.5.1	Monitor Active Red Channels	C.3.11.25	
			Monitor Active Red Channels
3.5.2.2.5.2	Monitor Active Yellow Channels	C.3.11.26	
			Monitor Active Yellow Channels
3.5.2.2.5.3	Monitor Active Green Channels	C.3.11.27	
			Monitor Active Green Channels
3.5.2.2.6	Retrieve Current Overlap Status Requirements		
3.5.2.2.6.1	Monitor Active Red Overlaps		
		C.3.11.28	Monitor Active Red Overlaps
3.5.2.2.6.2	Monitor Active Yellow Overlaps		
		C.3.11.29	Monitor Active Yellow Overlaps
3.5.2.2.6.3	Monitor Active Green Overlaps		
		C.3.11.30	Monitor Active Green Overlaps
3.5.2.2.6.4	Monitor Active Overlap Flashing Yellow Arrows		
		C.3.11.31	Monitor Active Overlap Flashing Yellow Arrows
3.5.2.2.6.5	Monitor Active Overlap Flashing Red Arrows		
		C.3.11.32	Monitor Active Overlap Flashing Red Arrows
3.5.2.2.7	Retrieve Current Preempt Status Requirements		
3.5.2.2.7.1	Monitor Currently Active Preempt		
		C.3.11.33	Monitor Currently Active Preempt
3.5.2.2.7.2	Monitor Current Preempt Inputs		
		C.3.11.34	Monitor Current Preempt Inputs
3.5.2.2.7.3	Monitor Current Preempt State		
		C.3.11.35	Monitor Current Preempt State
3.5.2.2.7.4	Monitor Current Gate Status		
		C.3.11.36	Monitor Current Gate Status
3.5.2.2.8	Retrieve Special Function Outputs Requirements		
3.5.2.2.8.1	Determine Maximum Number of Special Functions		
		C.3.11.37	Determine Maximum Number of Special Functions

3.5.2.2.8.2	Monitor Special Function Status		
		C.3.11.38	Monitor Special Function Status
3.5.2.2.8.3	Monitor Special Function Control Source		
		C.3.11.39	Monitor Special Function Control Source
3.5.2.2.9	Monitor Intra-Cabinet Communications Requirements		
3.5.2.2.9.1	Monitor TS2 Port 1 Status		
		C.3.9.13	Monitor TS2 Port 1 Status
3.5.2.2.9.2	Monitor TS2 Port 1 Fault Frame		
		C.3.9.14	Monitor TS2 Port 1 Fault Frame
3.5.2.2.9.3	Monitor ATC Serial Bus 1 Status		
		C.3.9.15	Monitor ATC Serial Bus 1 Status
3.5.2.2.10	Monitor Signal Monitoring Unit Requirements		
3.5.2.2.10.1	Monitor Signal Monitoring Unit Channel Voltage		
		C.3.9.16	Monitor Signal Monitoring Unit Channel Voltage
3.5.2.2.10.2	Monitor Signal Monitoring Unit Channel Current		
		C.3.9.17	Monitor Signal Monitoring Unit Channel Current
3.5.2.3	Manage Signal Operations Control Requirements		
3.5.2.3.1	Control ASC Function Requirements		
3.5.2.3.1.1	Enable/Disable Manual Backup		
		C.3.12.1	Enable/Disable Manual Backup
3.5.2.3.1.2	Control Global Minimum Recall		
		C.3.12.2	Control Global Minimum Recall
3.5.2.3.1.3	Control Call to Non-Actuated 1		
		C.3.12.3	Control Call to Non-Actuated 1
3.5.2.3.1.4	Control Call to Non-Actuated 2		
		C.3.12.4	Control Call to Non-Actuated 2
3.5.2.3.1.5	Control Walk Rest Modifier		
		C.3.12.5	Control Walk Rest Modifier
3.5.2.3.1.6	Control Interconnect		
		C.3.12.6	Control Interconnect

3.5.2.3.2	Command Timing Plan Requirements		
3.5.2.3.2.1	Activate System Timing Pattern Remotely		
		C.3.12.7	Activate System Timing Pattern Remotely
3.5.2.3.2.2	Control System Reference Point		
		C.3.12.8	Control System Reference Point
3.5.2.3.3	Control Phases Requirements		
3.5.2.3.3.1	Control Phase Omits		
		C.3.12.9	Control Phase Omits
3.5.2.3.3.2	Control Pedestrian Phase Omits		
		C.3.12.10	Control Pedestrian Phase Omits
3.5.2.3.3.3	Control Phase Holds		
		C.3.12.11	Control Phase Holds
3.5.2.3.3.4	Control Phase Force Offs		
		C.3.12.12	Control Phase Force Offs
3.5.2.3.3.5	Control Phase Vehicle Calls		
		C.3.12.13	Control Phase Vehicle Calls
3.5.2.3.3.6	Control Phase Pedestrian Calls		
		C.3.12.14	Control Phase Pedestrian Calls
3.5.2.3.4	Activate Preempt Remotely		
		C.3.12.15	Activate Preempt Remotely
3.5.2.3.5	Control Ring Requirements		
3.5.2.3.5.1	Control Ring Stop Time		
		C.3.12.16	Control Ring Stop Time
3.5.2.3.5.2	Control Ring Force Offs		
		C.3.12.17	Control Ring Force Offs
3.5.2.3.5.3	Control Ring Maximum 2 Time Settings		
		C.3.12.18	Control Ring Maximum 2 Time Settings
3.5.2.3.5.4	Control Ring Maximum 3 Time Settings		
		C.3.12.19	Control Ring Maximum 3 Time Settings
3.5.2.3.5.5	Control Ring Maximum Inhibit Settings		
		C.3.12.20	Control Ring Maximum Inhibit Settings
3.5.2.3.5.6	Control Ring Pedestrian Recycle Settings		
		C.3.12.21	Control Ring Pedestrian Recycle Settings

3.5.2.3.5.7	Control Ring Red Rest Settings		
		C.3.12.22	Control Ring Red Rest Settings
3.5.2.3.5.8	Control Ring Red Clearance Omit Settings		
		C.3.12.23	Control Ring Red Clearance Omit Settings
3.5.2.3.6	Activate Special Function Remotely		
		C.3.12.24	Activate Special Function Remotely
3.5.2.3.7	Remote Manual Control Requirements		
3.5.2.3.7.1	Enable Remote Manual Control		
		C.3.12.25	Remote Manual Control
3.5.2.3.7.2	Advance Interval During Remote Manual Control		
		C.3.12.25	Remote Manual Control
3.5.2.3.7.3	Configure Manual Control Timeout		
		C.3.12.25	Remote Manual Control
3.5.2.3.7.4	Enable/Disable Automatic Pedestrian Clearance Setting		
		C.3.12.26	Enable/Disable Automatic Pedestrian Clearance Setting
3.5.3	Detector Management Requirements		
3.5.3.1	Manage Detector Configuration Requirements		
3.5.3.1.1	Configure Vehicle Detector Requirements		
3.5.3.1.1.1	Configure Vehicle Detector Travel Mode		
		C.3.13.1	Configure Vehicle Detector Travel Mode
3.5.3.1.1.2	Configure Vehicle Detector Description		
		C.3.13.2	Configure Vehicle Detector Description
3.5.3.1.1.3	Configure Vehicle Detector Yellow Lock Call Enabled		
		C.3.13.3	Configure Vehicle Detector Yellow Lock Call Enabled
3.5.3.1.1.4	Configure Vehicle Detector Red Lock Call Enabled		
		C.3.13.4	Configure Vehicle Detector Red Lock Call Enabled
3.5.3.1.1.5	Configure Vehicle Detector Passage Enabled		

		C.3.13.5	Configure Vehicle Detector Passage Enabled
3.5.3.1.1.6	Configure Vehicle Detector Added Initial Time Enabled		
		C.3.13.6	Configure Vehicle Detector Added Initial Time Enabled
3.5.3.1.1.7	Configure Vehicle Detector Queue Enabled		
		C.3.13.7	Configure Vehicle Detector Queue Enabled
3.5.3.1.1.8	Configure Vehicle Detector Call Enabled		
		C.3.13.8	Configure Vehicle Detector Call Enabled
3.5.3.1.1.9	Configure Vehicle Detector Call Phase		
		C.3.13.9	Configure Vehicle Detector Call Phase
3.5.3.1.1.10	Configure Vehicle Detector Switch Phase		
		C.3.13.10	Configure Vehicle Detector Switch Phase
3.5.3.1.1.11	Configure Vehicle Detector Delay Time		
		C.3.13.11	Configure Vehicle Detector Delay Time
3.5.3.1.1.12	Configure Vehicle Detector Extend Time		
		C.3.13.12	Configure Vehicle Detector Extend Time
3.5.3.1.1.13	Configure Vehicle Detector Queue Limit Time		
		C.3.13.13	Configure Vehicle Detector Queue Limit Time
3.5.3.1.1.14	Configure Vehicle Detector No Activity Fault Time		
		C.3.13.14	Configure Vehicle Detector No Activity Fault Time
3.5.3.1.1.15	Configure Vehicle Detector Maximum Presence Time		
		C.3.13.15	Configure Vehicle Detector Maximum Presence Time
3.5.3.1.1.16	Configure Vehicle Detector Erratic Counts		
		C.3.13.16	Configure Vehicle Detector Erratic Counts
3.5.3.1.1.17	Configure Vehicle Detector Fail Time		
		C.3.13.17	Configure Vehicle Detector Fail Time
3.5.3.1.2	Configure Multiple Vehicle Detector Sets for Actuation		N/A
3.5.3.1.3	Configure Pedestrian Detector Requirements		

3.5.3.1.3.1	Configure Pedestrian Detector Description		
		C.3.13.18	Configure Pedestrian Detector Description
3.5.3.1.3.2	Configure Pedestrian Detector Call Phase		
		C.3.13.19	Configure Pedestrian Detector Call Phase
3.5.3.1.3.3	Configure Pedestrian Detector No Activity Fault Time		
		C.3.13.20	Configure Pedestrian Detector No Activity Fault Time
3.5.3.1.3.4	Configure Pedestrian Detector Maximum Presence Fault Time		
		C.3.13.21	Configure Pedestrian Detector Maximum Presence Fault Time
3.5.3.1.3.5	Configure Pedestrian Detector Erratic Counts		
		C.3.13.22	Configure Pedestrian Detector Erratic Counts
3.5.3.1.3.6	Configure Pedestrian Detector Non-Lock Calls		
		C.3.13.23	Configure Pedestrian Detector Non-Lock Calls
3.5.3.1.3.7	Configure Pedestrian Detector for Presence Detection		
		C.3.13.25	Configure Pedestrian Detector for Presence Detection
3.5.3.1.3.8	Configure Pedestrian Detector for Delayed Walk		
		C.3.13.26	Configure Pedestrian Detector for Delayed Walk
3.5.3.1.3.9	Configure Pedestrian Detector for Advanced Walk		
		C.3.13.27	Configure Pedestrian Detector for Advanced Walk
3.5.3.1.4	Configure Multiple Pedestrian Detector Sets for Actuation		N/A
3.5.3.1.5	Retrieve Detector Configuration Requirements		
3.5.3.1.5.1	Determine Maximum Number of Vehicle Detectors		
		C.3.13.28	Determine Maximum Number of Vehicle Detectors
3.5.3.1.5.2	Determine Maximum Number of Vehicle Detector Sets		

		C.3.13.29	Determine Maximum Number of Vehicle Detector Sets
3.5.3.1.5.3	Determine Maximum Number of Pedestrian Detectors		
		C.3.13.30	Determine Maximum Number of Pedestrian Detectors
3.5.3.1.5.4	Determine Maximum Number of Pedestrian Detector Sets		
		C.3.13.31	Determine Maximum Number of Pedestrian Detector Sets
3.5.3.2	Retrieve Detector Status Requirements		
3.5.3.2.1	Monitor Active Vehicle Detector Actuations		
		C.3.13.32	Monitor Active Vehicle Detector Actuations
3.5.3.2.2	Monitor Active Pedestrian Detector Actuations		
		C.3.13.33	Monitor Active Pedestrian Detector Actuations
3.5.3.3	Retrieve Detector Health Requirements		
3.5.3.3.1	Retrieve Vehicle Detector Health Requirements		
3.5.3.3.1.1	Monitor Vehicle Detector Alarm Status		
		C.3.13.34	Monitor Vehicle Detector Alarm Status
3.5.3.3.1.2	Monitor Vehicle Detector Faults from Controller		
		C.3.13.35	Monitor Vehicle Detector Faults from Controller
3.5.3.3.1.3	Monitor Vehicle Detector Faults from Detector		
		C.3.13.36	Monitor Vehicle Detector Faults from Detector
3.5.3.3.2	Retrieve Pedestrian Detector Health Requirements		
3.5.3.3.2.1	Monitor Pedestrian Detector Alarm Status		
		C.3.13.37	Monitor Pedestrian Detector Alarm Status
3.5.3.3.2.2	Monitor Pedestrian Detector Faults		
		C.3.13.38	Monitor Pedestrian Detector Faults
3.5.3.4	Control Detector Requirements		
3.5.3.4.1	Control Vehicle Detector Reset		

		C.3.13.39	Control Vehicle Detector Reset
3.5.3.4.2	Control Pedestrian Detector Reset		
		C.3.13.40	Control Pedestrian Detector Reset
3.5.3.4.3	Control Detector Diagnostic Reset		
		C.3.13.41	Control Detector Diagnostic Reset
3.5.3.4.4	Control Vehicle Detector Actuation		
		C.3.13.42	Control Vehicle Detector Actuation
3.5.3.4.5	Control Pedestrian Detector Actuation		
		C.3.13.43	Control Pedestrian Detector Actuation
3.5.3.5	Manage Detector Data Collection Requirements		
3.5.3.5.1	Monitor Vehicle Detector Data Requirements		
3.5.3.5.1.1	Monitor Vehicle Detector Data Sequence		
		C.3.14.1	Monitor Vehicle Detector Data Sequence
3.5.3.5.1.2	Monitor Vehicle Volume Data		
		C.3.14.2	Monitor Vehicle Volume Data
3.5.3.5.1.3	Monitor Vehicle Occupancy Data		
		C.3.14.3	Monitor Vehicle Occupancy Data
3.5.3.5.1.4	Monitor Vehicle Average Speed		
		C.3.14.4	Monitor Vehicle Average Speed
3.5.3.5.1.5	Monitor Vehicle Detector Data Sample Time		
		C.3.14.5	Monitor Vehicle Detector Data Sample Time
3.5.3.5.1.6	Monitor Vehicle Detector Data Sample Duration		
		C.3.14.6	Monitor Vehicle Detector Data Sample Duration
3.5.3.5.2	Monitor Pedestrian Detector Data Requirements		
3.5.3.5.2.1	Monitor Pedestrian Detector Data Sequence		
		C.3.14.7	Monitor Pedestrian Detector Data Sequence
3.5.3.5.2.2	Monitor Pedestrian Counts		
		C.3.14.8	Monitor Pedestrian Counts
3.5.3.5.2.3	Monitor Pedestrian Actuations		
		C.3.14.9	Monitor Pedestrian Actuations

3.5.3.5.2.4	Monitor Pedestrian Services		
		C.3.14.10	Monitor Pedestrian Services
3.5.3.5.2.5	Monitor Pedestrian Detector Data Sample Time		
		C.3.14.11	Monitor Pedestrian Detector Data Sample Time
3.5.3.5.2.6	Monitor Pedestrian Detector Data Sample Duration		
		C.3.14.12	Monitor Pedestrian Detector Data Sample Duration
3.5.3.5.3	Configure Detector Data Collection Requirements		
3.5.3.5.3.1	Configure Vehicle Detector Data Sample Period		
		C.3.14.13	Configure Vehicle Detector Data Sample Period
3.5.3.5.3.2	Configure Pedestrian Detector Data Sample Period		
		C.3.14.14	Configure Pedestrian Detector Data Sample Period
3.5.3.5.3.3	Configure Vehicle Speed Detectors		
		C.3.14.15	Configure Vehicle Speed Detectors
3.5.3.5.3.4	Configure Single Detector Speed Mode		
		C.3.14.16	Configure Single Detector Speed Mode
3.5.3.5.3.5	Configure Paired Detector		
		C.3.14.17	Configure Paired Detector
3.5.3.5.3.6	Configure Paired Detector Placement		
		C.3.14.18	Configure Paired Detector Placement
3.5.3.5.3.7	Configure Paired Detector Spacing		
		C.3.14.19	Configure Paired Detector Spacing
3.5.3.5.3.8	Configure Average Vehicle Length		
		C.3.14.20	Configure Average Vehicle Length
3.5.3.5.3.9	Configure Vehicle Detection Zone Length		
		C.3.14.21	Configure Vehicle Detection Zone Length
3.3.3.5.4	Configure Multiple Vehicle Detector Sets for Data Collection		N/A
3.5.4	Connected Vehicles Interface Management		

3.5.4.1	Manage ASC - RSU Interface Requirements		
3.5.4.1.1	Configure ASC Communications Port for RSU		
		C.3.15.1	Configure ASC Communications Port for RSU
3.5.4.1.2	Configure Logical RSU Ports and Address		
		C.3.15.2	Configure Logical RSU Ports and Address
3.5.4.1.3	Configure RSU Interface Polling Period		
		C.3.15.3	Configure RSU Interface Polling Period
3.5.4.1.4	Configure RSU Interface Watchdog		
		C.3.15.4	Configure RSU Interface Watchdog
3.5.4.1.5	Monitor RSU Interface Watchdog Timer		
		C.3.15.5	Monitor RSU Interface Watchdog Timer
3.5.4.1.6	Monitor RSU Interface Watchdog Alarm		
		C.3.11.3	Monitor Unit Level Alarms
3.5.4.2	Manage ASC Process Requirements		
3.5.4.2.1	Manage Signal Phase and Timing Requirements		
3.5.4.2.1.1	Enable Signal Phase and Timing Data	C.3.15.6	
			Enable Signal Phase and Timing Data
3.5.4.2.1.2	Retrieve Signal Phase and Timing Generation Time		
		C.3.15.7	Retrieve Signal Phase and Timing Generation Time
3.5.4.2.1.3	Exchange Movement Status Requirements		
3.5.4.2.1.3.1	Retrieve Movement Timing Requirements		
3.5.4.2.1.3.1.1	Monitor Movement Minimum End Time		
		C.3.15.8	Monitor CV Movement Minimum End Time
3.5.4.2.1.3.1.2	Monitor Movement Maximum End Time		
		C.3.15.9	Monitor CV Movement Maximum End Time
3.5.4.2.1.3.1.3	Monitor Movement Likely End Time		
		C.3.15.10	Monitor CV Movement Expected End Time

3.5.4.2.1.3.1.4	Monitor Movement Likely End Time Confidence		
		C.3.15.11	Monitor CV Movement Likely End Time Confidence
3.5.4.2.1.3.1.5	Monitor Movement Next Occurrence		
		C.3.15.12	Monitor CV Movement Next Occurrence
3.5.4.2.1.3.1.6	Monitor Movement Start Time		N/A
3.5.4.2.1.3.1.7	Monitor Next Movement Minimum End Time		
		C.3.15.13	Monitor CV Next Movement Minimum End Time
3.5.4.2.1.3.1.8	Monitor Next Movement Maximum End Time		
		C.3.15.14	Monitor CV Next Movement Maximum End Time
3.5.4.2.1.3.1.9	Monitor Next Movement Start Time		
		C.3.15.15	Monitor Next CV Movement Start Time
3.5.4.2.1.3.1.10	Determine Maximum Number of Movement Events		
		C.3.15.16	Determine Maximum Number of CV Movement Events
3.5.4.2.1.3.2	Configure Movement Assistance Requirements		
3.5.4.2.1.3.2.1	Configure Queue Detectors for Movement Assistance		
		C.3.15.17	Configure Queue Detectors for CV Movement Assistance
3.5.4.2.1.3.2.2	Configure Pedestrian Detectors for Movement Conflict Assistance		
		C.3.15.18	Configure Pedestrian Detectors for CV Movement Conflict Assistance
3.5.4.2.1.3.2.3	Configure Bicycle Detectors for Movement Conflict Assistance		
		C.3.15.19	Configure Bicycle Detectors for CV Movement Conflict Assistance
3.5.4.2.1.3.3	Retrieve Movement Assistance Requirements		
3.5.4.2.1.3.3.1	Monitor Lane Connection Queue Length		
		C.3.15.20	Monitor CV Lane Connection Queue Length
3.5.4.2.1.3.3.2	Monitor Lane Connection Vulnerable Road User Detection		

		C.3.15.21	Monitor CV Lane Connection Vulnerable Road User Detection
3.5.4.2.1.3.4	Manage Advisory Speed Requirements		
3.5.4.2.1.3.4.1	Configure Advisory Speed Type		
		C.3.15.22	Configure Advisory Speed Type
3.5.4.2.1.3.4.2	Configure Advisory Speed		
		C.3.15.23	Configure Advisory Speed
3.5.4.2.1.3.4.3	Configure Advisory Speed Zone		
		C.3.15.24	Configure Advisory Speed Zone
3.5.4.2.1.3.4.4	Configure Advisory Speed Vehicle Type		
		C.3.15.25	Configure Advisory Speed Vehicle Type
3.5.4.2.1.3.5	Monitor Movement State		
		C.3.15.26	Monitor CV Movement States
3.5.4.2.1.3.6	Monitor Next Movement State		
		C.3.15.26	Monitor CV Movement States
3.5.4.2.1.3.7	Monitor Movement Status		
		C.3.15.27	Monitor CV Movement Status
3.5.4.2.1.4	Manage Enabled Lane Requirements		
3.5.4.2.1.4.1	Configure Concurrent Enabled Lanes		
		C.3.15.28	Configure Concurrent Enabled Lanes
3.5.4.2.1.4.2	Configure Enabled Lanes by Time of Day		
		C.3.15.29	Configure Enabled Lanes by Time of Day
3.5.4.2.1.4.3	Determine Lanes Enabled		
		C.3.15.30	Determine Lanes Enabled
3.5.4.2.1.4.4	Command Enabled Lanes		N/A
3.5.4.2.1.5	Enable Signal Phase and Timing Exchange		
		C.3.15.31	Enable Signal Phase and Timing Data Exchange by Port
3.5.4.2.1.6	Configure Road Authority Identifier		
		C.3.15.32	Configure Road Authority Identifier
3.5.4.2.1.7	Retrieve Signal Phase and Timing Intersection Status Requirements		
3.5.4.2.1.7.1	Monitor Manual Control Indication		

		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.2	Monitor Stop Indication		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.3	Monitor Failure Flash Indication		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.4	Monitor Preemption Operation Indication		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.5	Monitor Priority Operation Indication		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.6	Monitor Fixed Time Control Indication		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.7	Monitor Non-Fixed Time Control Indication		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.8	Monitor Standby Operation Indication		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.9	Monitor Controller Failure		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.10	Monitor MAP Message Validity		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.7.11	Monitor SPaT Data Validity		
		C.3.15.33	Retrieve Signal Phase and Timing Data Intersection Status
3.5.4.2.1.8	Mark SPaT Invalid - Controller		
		C.3.15.34	Mark SPaT Invalid - Controller
3.5.4.2.1.9	Mark SPaT Invalid - Port		
		C.3.15.35	Mark SPaT Invalid - Port
3.5.4.2.1.10	Mark MAP Message Invalid - Controller		
		C.3.15.36	Mark MAP Message Invalid - Controller

3.5.4.2.1.11	Mark MAP Message Invalid - Port		
		C.3.15.37	Mark MAP Message Invalid - Port
3.5.4.2.1.12	Manage Signal Group Requirements		
3.5.4.2.1.12.1	Determine Maximum Number of Signal Groups		
		C.3.15.38	Determine Maximum Number of Signal Groups
3.5.4.2.1.12.2	Configure Signal Group Intersection Mapping		
		C.3.15.39	Configure Signal Group Intersection Mapping
3.5.4.2.1.12.3	Configure Signal Group Control Source		
		C.3.15.40	Configure Signal Group Control Source
3.5.4.2.1.12.4	Configure Signal Group Indication Types		
		C.3.15.41	Configure Signal Group Indication Types
3.5.4.2.1.12.5	Configure Signal Group Protected or Permissive State		
		C.3.15.42	Configure Signal Group Protected or Permissive State
3.5.4.2.1.12.6	Configure Signal Group Revocable Lanes		
		C.3.15.43	Configure Signal Group Revocable Lanes
3.5.4.2.1.12.7	Determine Maximum Number of Signal State Entries		
		C.3.15.44	Determine Maximum Number of Signal State Entries
3.5.4.2.1.12.8	Configure Customized Signal State Parameters		
		C.3.15.45	Configure Customized Signal State Parameters
3.5.4.2.1.13	Retrieve Signal Phase and Timing Time Point		
		C.3.15.46	Retrieve Signal Phase and Timing Time Point
3.5.4.2.2	Manage Connected Vehicle Detector Requirements		
3.5.4.2.2.1	Enabled Connected Device Connection		N/A
3.5.4.2.2.2	Configure Vehicle Detector for Connected Vehicle Applications		N/A
3.5.4.2.2.3	Configure Connected Vehicle Detector Input Assignment		N/A

3.5.4.2.2.4	Configure Connected Vehicle Detector Port Assignment		N/A
3.5.4.2.2.5	Configure Assured Green Period Duration		N/A
3.5.4.2.2.6	Configure Red Light Violation Warning Application Parameters		N/A
3.5.4.3	Manage ASC – CV Application Process Interface Requirements		
3.5.4.3.1	ASC - External CV Application Process Requirements		
3.5.4.3.1.1	Provide Movement Information Requirements		
3.5.4.3.1.1.1	Provide Movement Time Point		N/A
3.5.4.3.1.1.2	Provide Movement State		N/A
3.5.4.3.1.1.3	Provide Movement Minimum End Time		N/A
3.5.4.3.1.1.4	Provide Movement Maximum End Time		N/A
3.5.4.3.1.1.5	Provide Movement Likely End Time		N/A
3.5.4.3.1.1.5	Provide Movement Likely End Time		N/A
3.5.4.3.1.1.6	Provide Movement Likely End Time Confidence		N/A
3.5.4.3.1.1.7	Provide Next Movement State		N/A
3.5.4.3.1.1.8	Provide Next Movement Minimum End Time		N/A
3.5.4.3.1.1.9	Provide Next Movement Maximum End Time		N/A
3.5.4.3.1.1.10	Provide Next Movement Start Time		N/A
3.5.4.3.1.2	Provide Movement Assistance Requirements		
3.5.4.3.1.2.1	Provide Lane Connection Queue Length		N/A
3.5.4.3.1.2.2	Provide Lane Connection Vulnerable Road User Detection		N/A
3.5.4.3.1.3	Provide Advisory Speed Requirements		
3.5.4.3.1.3.1	Provide Advisory Speed Type		N/A
3.5.4.3.1.3.2	Provide Advisory Speed		N/A

3.5.4.3.1.3.3	Provide Advisory Speed Zone		N/A
3.5.4.3.1.3.4	Provide Advisory Speed Vehicle Type		N/A
3.5.4.3.1.4	Provide Road Authority ID		N/A
3.5.4.3.1.5	Provide Signal Phase and Timing Intersection Status		N/A
3.5.4.3.1.6	Provide Compressed SPaT Information to External CV Application Process		N/A
3.5.4.3.2	ASC - Internal CV Application Process Requirements		
3.5.4.3.2.1	Provide UPER-encoded SPaT Message		N/A
3.5.4.3.2.2	Retrieve BSMs		N/A
3.5.4.3.2.3	Retrieve PSMs		N/A
3.5.4.3.2.4	Retrieve Actuation Report		N/A
3.5.4.3.2.5	Retrieve Detection Report		N/A
3.5.4.3.3	Exchange Roadway Geometrics Information Requirements		
3.5.4.3.3.1	Retrieve MAP Plan in Effect		
		C.3.15.47	Retrieve MAP Plan in Effect
3.5.4.3.3.2	Confirm MAP Plan Compatibility		N/A
3.5.4.3.4	Monitor CV Certificate Faults		
		C.3.11.3	Monitor Unit Level Alarms
3.5.4.4	Manage ASC - ECLA Interface Requirements		
3.5.4.4.1	Receive Current Phase Minimum End Time from an ECLA		N/A
3.5.4.4.2	Receive Current Phase Maximum End Time from an ECLA		N/A
3.5.4.4.3	Receive Current Phase Likely End Time from an ECLA		N/A
3.5.4.4.4	Receive Current Phase Likely End Time Confidence from an ECLA		N/A
3.5.4.4.5	Receive Next Phase from an ECLA		N/A
3.5.4.4.6	Receive Compressed ECLA Input Data		N/A
3.5.5	Backward Compatibility Requirements		

3.6	Supplemental Non-communications Requirements		
3.6.1	Response Time for Requests		N/A
3.6.2	Condition-based Maximum Start Time		N/A
3.6.3	Signal Phase and Timing Data Performance Requirements		
3.6.3.1	SPaT Maximum Transmission Start Time		N/A
3.6.3.2	Movement Time Point Minimum Transmission Rate		N/A
3.6.3.3	SPaT Maximum Transmission Rate		N/A
3.6.3.4	SPaT Time Accuracy		N/A

C.3 Test Procedures

C.3.1 System Information

C.3.1.1 Create Database Transaction

Test Procedure:	**Create Database Transaction**
Description:	**This test case has the ASC perform consistency checks.**
Requirement(s):	
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: fdTransactionMode.	Pass/Fail
2	IF fdTransactionMode IS_NOT_EQUAL_TO 1.	
2.1	NOTE '1 = normal, 4 = done'.	
2.2	GOTO step 1.	
3	ASSIGN fdTransactionMode EQUALS 2.	
3.1	NOTE '2 = transaction'.	
4	SET the following objects: fdTransactionMode.	Pass/Fail
5	GET the following objects: fdTransactionMode.	Pass/Fail
6	IF fdTransactionMode IS_NOT_EQUAL_TO 2.	

6.1	GOTO step 5.	
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.1.2 Verify Database Status and Verify Database Error

Test Procedure:	**C.3.1.2 Verify Database Status and Verify Database Error**
Description:	**This test case has a management station monitor the results of a consistency check.**
Requirement(s):	
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. If step 8.3 is not executed, the test case that called this test case is considered failed.**

Test Step Number	Test Procedure	Results
1	ASSIGN fdTransactionMode EQUALS 3.	
1.1	NOTE '3 = verify'.	
2	SET the following objects: fdTransactionMode.	Pass/Fail
3	DELAY 500 millisecond(s).	
4	GET the following objects: fdTransactionMode.	Pass/Fail
5	VERIFY fdTransactionMode IS_EQUAL_TO 4.	Pass/Fail
5.1	NOTE '4 = done'.	
6	GET the following objects: fdTransactionStatus.	Pass/Fail
7	IF fdTransactionStatus IS_EQUAL_TO 3.	
7.1	GOTO step '6'.	
7.2	NOTE '3 = verifying'.	
8	IF fdTransactionStatus IS_EQUAL_TO 4.	
8.1	ASSIGN fdTransactionMode EQUALS 1.	
8.2	SET the following objects: fdTransactionMode.	Pass/Fail
8.3	EXIT	
8.4	NOTE 'For dbVerifyStatus, 4 = doneWithNoError'.	
8.5	NOTE 'For fdTransactionMode, 1 = normal'.	
10	IF fdTransactionStatus IS_EQUAL_TO 5.	
10.1	GET the following objects: fdTransactionError.	Pass/Fail
10.2	NOTE '5 = doneWithError'.	
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.1.3 Explore Data

Test Procedure:	Explore Data	
Description:	This test case verifies that the ASC allows a management station to configure which Flashing Arrows consists of overlaps.	
Requirement(s):		
Variable(s):	Last_Object_OID RESPONSE ERROR Continue	OBJECT IDENTIFIER Int Boolean
Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.	

Test Step Number	Test Procedure	Results
1	ASSIGN Last_Object_OID EQUALs ZeroDotZero	
2	GET-NEXT the following variables: Last_Object_OID.	
3	IF RESPONSE ERROR IS_NOT_EQUAL_TO noError(0).	
3.1	VERIFY RESPONSE ERROR IS_EQUAL_TO NoSuchObject.	Pass/Fail
4	USER-ACTION 'Determine whether the OID of the returned object is lexicographically larger than the OID contained in the request.'	
4.1	RECORD this information as Continue.	
5	IF RESPONSE ERROR IS_EQUAL_TO noError(0).	
5.1	VERIFY Continue IS_EQUAL_TO 1.	Pass/Fail
5.2	GOTO step 7.	
6	USER-ACTION 'Verify that the OID of the retrieved object is identical to the OID contained in the request.'	
7	USER-ACTION 'Determine the OID value of the retrieved object.'	
7.1	RECORD this information as Last_Object_OID.	
8	IF RESPONSE ERROR IS_EQUAL_TO noError(0).	
8.1	IF Continue IS_EQUAL_TO 1.	
8.1.1	GOTO step 2.	
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.1.4 Monitor SNMP Information

Test Procedure:	**Monitor SNMP Information**
Description:	**This test case verifies that the ASC allows a management station to retrieve basic instrumentation and control information of the SNMP device.**
Requirement(s):	• **3.4.1.4.1 Monitor SNMP Information**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: snmpInPkts, snmpInBadVersions, snmpInBadCommunityNames, snmpInBadCommunityUses, snmpInASNParseErrs, snmpSilentDrops, snmpProxyDrops	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.2 Unit Settings

C.3.2.1 Configure Start-Up Flash Mode

Test Procedure:	**Configure Startup Flash Mode**
Description:	**This test case verifies that the ASC allows a management station to configure how the ASC flashes during startup.**
Requirement(s):	• **3.5.2.1.1.1.1 Configure Startup Flash Mode**
Variable(s):	**OriginalFlashMode** **unitStartUpFlashMode** **TestFlashMode** **unitStartUpFlashMode**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitStartUpFlashMode.	Pass/Fail
1.1	RECORD this information as OriginalFlashMode.	
2	ASSIGN TestFlashMode EQUALS RANDOM (1 TO 3).	
2.1	IF TestFlashMode IS_EQUAL_TO OriginalFlashMode.	
2.1.1	GOTO step 2.	
3	ASSIGN unitStartUpFlashMode EQUALS TestFlashMode.	
4	SET the following objects: unitStartUpFlashMode.	Pass/Fail
5	GET the following objects: unitStartUpFlashMode.	Pass/Fail
6	VERIFY unitStartUpFlashMode IS_EQUAL_TO TestFlashMode.	Pass/Fail
7	ASSIGN unitStartUpFlashMode EQUALS OriginalFlashMode.	
8	SET the following objects: unitStartUpFlashMode.	Pass/Fail
9	GET the following objects: unitStartUpFlashMode.	Pass/Fail
10	VERIFY unitStartUpFlashMode IS_EQUAL_TO OriginalFlashMode.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.2.2 Configure Start-Up Flash Mode

Test Procedure:	**Configure Startup Flash Time**
Description:	**This test case verifies that the ASC allows a management station to configure the startup flash time.**
Requirement(s):	• **3.4.1.2 Deliver Data** • **3.5.2.1.1.1.2 Configure Startup Flash Time**
Variable(s):	**OriginalStartUpFlash** **unitStartUpFlash** **TestStartUpFlash** **unitStartUpFlash**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitStartUpFlash.	Pass/Fail
1.1	RECORD this information as OriginalStartUpFlash.	
2	ASSIGN TestStartUpFlash EQUALS RANDOM (0 TO 255).	
2.1	IF TestStartUpFlash IS_EQUAL_TO OriginalStartUpFlash .	
2.1.1	GOTO step 2.	
3	ASSIGN unitStartUpFlash EQUALS TestStartUpFlash.	
4	SET the following objects: unitStartUpFlash.	Pass/Fail
5	GET the following objects: unitStartUpFlash.	Pass/Fail
6	VERIFY unitStartUpFlash IS_EQUAL_TO TestStartUpFlash.	Pass/Fail
7	ASSIGN unitStartUpFlash EQUALS OriginalStartUpFlash.	
8	SET the following objects: unitStartUpFlash.	Pass/Fail
9	GET the following objects: unitStartUpFlash.	Pass/Fail
10	VERIFY unitStartUpFlash IS_EQUAL_TO OriginalStartUpFlash.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.2.3 Configure Backup Time

Test Procedure:	**Configure Backup Time**
Description:	**This test case verifies that the ASC allows a management station to configure the backup time.**
Requirement(s):	• **3.5.2.1.1.2 Configure Backup Time**
Variable(s):	**OriginalBackUpTime unitBackupTime** **TestBackUpTime unitBackupTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitBackupTime.	Pass/Fail
1.1	RECORD this information as OriginalBackUpTime.	
2	ASSIGN TestBackUpTime EQUALS RANDOM (0 TO 16777216).	
2.1	IF TestBackUpTime IS_EQUAL_TO OriginalBackUpTime.	
2.1.1	GOTO step 2.	
3	ASSIGN unitBackupTime EQUALS TestBackUpTime.	
4	SET the following objects: unitBackupTime.	Pass/Fail
5	GET the following objects: unitBackupTime.	Pass/Fail
6	VERIFY unitBackupTime IS_EQUAL_TO TestBackUpTime.	Pass/Fail
7	ASSIGN unitBackupTime EQUALS OriginalBackUpTime.	

8	SET the following objects: unitBackupTime.	Pass/Fail
9	GET the following objects: unitBackupTime.	Pass/Fail
10	VERIFY unitBackupTime IS_EQUAL_TO OriginalBackUpTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.2.4 Configure ASC Antenna Offset

Test Procedure:	**Configure ASC Antenna Offset**
Description:	**This test case verifies that the ASC allows a management station to configure an offset between the antenna of a mounted external GNSS positioning device and the base of the ASC.**
Requirement(s):	• **3.5.1.1.1 Configure ASC Location – Antenna Offset**
Variable(s):	**OriginalElevationOffset** **ascElevationOffset** **TestElevationOffset** **ascElevationOffset**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: ascElevationOffset.	Pass/Fail
1.1	RECORD this information as OriginalElevationOffset.	
2	ASSIGN TestElevationOffset EQUALS RANDOM (0 TO 31).	
2.1	IF TestElevationOffset IS_EQUAL_TO OriginalElevationOffset	
2.1.1	GOTO Step 2	
3	ASSIGN ascElevationOffset EQUALS TestElevationOffset.	
4	SET the following objects: ascElevationOffset.	Pass/Fail
5	GET the following objects: ascElevationOffset.	Pass/Fail
6	VERIFY TestElevation IS_EQUAL_TO TestElevationOffset.	Pass/Fail
7	ASSIGN ascElevationOffset EQUALS OriginalElevationOffset.	
8	SET the following objects: ascElevationOffset.	Pass/Fail
9	GET the following objects: ascElevationOffset.	Pass/Fail
10	VERIFY TestElevation IS_EQUAL_TO OriginalElevationOffset.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3 Phases

C.3.3.1 Enable/Disable Phase

Test Procedure:	**Enable/Disable Phase**	
Description:	**This test case verifies that the ASC shall allow a management station to enable/disable phases.**	
Requirement(s):	• **3.5.2.1.2.1.1 Enable/Disable Phase**	
Variable(s):	**Table_Row** **OriginalPhaseOptions** **TestPhaseOptions** **MaxRows**	**Int** **phaseOptions** **phaseOptions** **maxPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 1.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.2 Configure Phase Minimum Green Time

Test Procedure:	**Configure Phase Minimum Green Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the minimum green time for a phase.**	
Requirement(s):	• **3.5.2.1.2.1.2 Configure Phase Minimum Green Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseMinimumGreen** **TestPhaseMinimumGreen**	**maxPhases** **Int** **phaseMinimumGreen** **phaseMinimumGreen**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseMinimumGreen.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseMinimumGreen.	
4	ASSIGN TestPhaseMinimumGreen EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseMinimumGreen IS_EQUAL_TO OriginalPhaseMinimumGreen.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseMinimumGreen EQUALS TestPhaseMinimumGreen.	
6	SET the following objects: phaseMinimumGreen.Table_Row.	Pass/Fail
7	GET the following objects: phaseMinimumGreen.Table_Row.	Pass/Fail
8	VERIFY phaseMinimumGreen.Table_Row IS_EQUAL_TO TestPhaseMinimumGreen.	Pass/Fail
9	ASSIGN phaseMinimumGreen.Table_Row EQUALS OriginalPhaseMinimumGreen.	
10	SET the following objects: phaseMinimumGreen.Table_Row.	Pass/Fail
11	GET the following objects: phaseMinimumGreen.Table_Row.	Pass/Fail
12	VERIFY phaseMinimumGreen.Table_Row IS_EQUAL_TO OriginalPhaseMinimumGreen.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.3 Configure Phase Passage Time

	Configure Phase Passage Time

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to configure the passage timer for the phase. This is the time the phase's GREEN indication will be extended when any vehicle actuation is removed.**
Requirement(s):	• **3.5.2.1.2.1.3 Configure Phase Passage Time**
Variable(s):	**Table_Row** **Int** **MaxRows** **maxPhases** **OriginalPhasePassage** **phasePassage** **TestPhasePassage** **phasePassage**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phasePassage.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhasePassage.	
4	ASSIGN TestPhasePassage EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhasePassage IS_EQUAL_TO OriginalPhasePassage.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePassage.Table_Row EQUALS TestPhasePassage.	
6	SET the following objects: phasePassage.Table_Row.	Pass/Fail
7	GET the following objects: phasePassage.Table_Row.	Pass/Fail
8	VERIFY phasePassage.Table_Row IS_EQUAL_TO TestPhasePassage.	Pass/Fail
9	ASSIGN phasePassage.Table_Row EQUALS OriginalPhasePassage.	
10	SET the following objects: phasePassage.Table_Row.	Pass/Fail
11	GET the following objects: phasePassage.Table_Row.	Pass/Fail
12	VERIFY phasePassage.Table_Row IS_EQUAL_TO OriginalPhasePassage.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.4 Configure Two Phase Maximum Green Times

Test Procedure:	**Configure Two Phase Maximum Green Times**
Description:	**This test case verifies that the ASC allows a management station to configure the maximum length of time the phase display a GREEN indication.**

Requirement(s):	• **3.5.2.1.2.1.4 Configure Two Phase Maximum Green Times**
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **TestPhaseMaximum1** **phaseMaximum1** **OriginalPhaseMaximum1** **phaseMaximum1** **TestPhaseMaximum2** **phaseMaximum2** **OriginalPhaseMaximum2** **phaseMaximum1**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseMaximum1.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseMaximum1.	
4	GET the following objects: phaseMaximum2.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseMaximum2.	
5	ASSIGN TestPhaseMaximum1 EQUALS RANDOM (0 TO 999).	
5.1	IF TestPhaseMaximum1 IS_EQUAL_TO OriginalPhaseMaximum1.	
5.1.1	GOTO step 5.	
6	ASSIGN TestPhaseMaximum2 EQUALS RANDOM (0 TO 999).	
6.1	IF TestPhaseMaximum2 IS_EQUAL_TO OriginalPhaseMaximum2.	
6.1.1	GOTO step 6.	
7	ASSIGN phaseMaximum1.Table_Row EQUALS TestPhaseMaximum1.	
8	ASSIGN phaseMaximum2.Table_Row EQUALS TestPhaseMaximum2.	
9	SET the following objects: phaseMaximum1.Table_Row, phaseMaximum2.Table_Row.	Pass/Fail
10	GET the following objects: phaseMaximum1.Table_Row, phaseMaximum2.Table_Row.	Pass/Fail
11	VERIFY phaseMaximum1.Table_Row IS_EQUAL_TO TestPhaseMaximum1.	Pass/Fail
12	VERIFY phaseMaximum2.Table_Row IS_EQUAL_TO TestPhaseMaximum2.	Pass/Fail
13	ASSIGN phaseMaximum1.Table_Row EQUALS OriginalPhaseMaximum1.	
14	ASSIGN phaseMaximum2.Table_Row EQUALS OriginalPhaseMaximum2.	
15	SET the following objects: phaseMaximum1.Table_Row, phaseMaximum2.Table_Row.	Pass/Fail
16	GET the following objects: phaseMaximum1.Table_Row, phaseMaximum2.Table_Row.	Pass/Fail
17	VERIFY phaseMaximum1.Table_Row IS_EQUAL_TO OriginalPhaseMaximum1.	Pass/Fail
18	VERIFY phaseMaximum2.Table_Row IS_EQUAL_TO OriginalPhaseMaximum2.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.5 Configure Three Phase Maximum Green Times

Test Procedure:	**Configure Three Phase Maximum Green Times**
Description:	**This test case verifies that the ASC allows a management station to configure three maximum times a phase may be held in Green.**
Requirement(s):	• **3.5.2.1.2.1.5 Configure Three Phase Maximum Green Times**
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalPhaseMaximum3** phaseMaximum3 **TestPhaseMaximum3** phaseMaximum3
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PERFORM the Test Procedure 'C.3.3.4 Configure Two Phase Maximum Green Times'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
4	GET the following objects: phaseMaximum3.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseMaximum3.	
5	ASSIGN TestThreshold EQUALS RANDOM (0 TO 999).	
5.1	IF TestPhaseMaximum3 IS_EQUAL_TO OriginalPhaseMaximum3.	
5.1.1	GOTO step 4.	
6	ASSIGN phaseMaximum3.Table_Row EQUALS TestPhaseMaximum3.	
7	SET the following objects: phaseMaximum3.Table_Row.	Pass/Fail
8	GET the following objects: phaseMaximum3.Table_Row.	Pass/Fail
9	VERIFY phaseMaximum3.Table_Row IS_EQUAL_TO TestPhaseMaximum3.	Pass/Fail
10	ASSIGN phaseMaximum3.Table_Row EQUALS OriginalPhaseMaximum3.	
11	SET the following objects: phaseMaximum3.Table_Row.	Pass/Fail
12	GET the following objects: phaseMaximum3.Table_Row.	Pass/Fail
13	VERIFY phaseMaximum3.Table_Row IS_EQUAL_TO OriginalPhaseMaximum3.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.3.6 Configure Phase Yellow Change Time

Test Procedure:	**Configure Phase Yellow Change Time**
Description:	**This test case verifies that the ASC allows a management station to configure the yellow change interval time for a phase.**
Requirement(s):	• **3.5.2.1.2.1.6 Configure Phase Yellow Change Time**
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalPhaseYellowChange** phaseYellowChange **TestYellowPhaseChange** phaseYellowChange
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseYellowChange.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseYellowChange.	
4	ASSIGN TestYellowPhaseChange EQUALS RANDOM (0 TO 255).	
4.1	IF TestYellowPhaseChange IS_EQUAL_TO OriginalPhaseYellowChange.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseYellowChange.Table_Row EQUALS TestYellowPhaseChange.	
6	SET the following objects: phaseYellowChange.Table_Row.	Pass/Fail
7	GET the following objects: phaseYellowChange.Table_Row.	Pass/Fail
8	VERIFY phaseYellowChange.Table_Row IS_EQUAL_TO TestYellowPhaseChange.	Pass/Fail
9	ASSIGN phaseYellowChange.Table_Row EQUALS OriginalPhaseYellowChange.	
10	SET the following objects: phaseYellowChange.Table_Row.	Pass/Fail
11	GET the following objects: phaseYellowChange.Table_Row.	Pass/Fail
12	VERIFY phaseYellowChange.Table_Row IS_EQUAL_TO OriginalPhaseYellowChange.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.3.7 Configure Phase Red Clearance Time

Test Procedure:	**Configure Phase Red Clearance Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the red clearance interval time for a phase.**	
Requirement(s):	• **3.5.2.1.2.1.7 Configure Red Clearance Time**	
Variable(s):	**MaxRows** **OriginalPhaseRedClear** **TestPhaseRedClear** **Table_Row**	**maxPhases** **phaseRedClear** **phaseRedClear** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseRedClear.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseRedClear.	
4	ASSIGN TestPhaseRedClear EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseRedClear IS_EQUAL_TO OriginalPhaseRedClear.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseRedClear.Table_Row EQUALS TestPhaseRedClear.	
6	SET the following objects: phaseRedClear.Table_Row.	Pass/Fail
7	GET the following objects: phaseRedClear.Table_Row.	Pass/Fail
8	VERIFY phaseRedClear.Table_Row IS_EQUAL_TO TestPhaseRedClear.	Pass/Fail
9	ASSIGN phaseRedClear.Table_Row EQUALS OriginalPhaseRedClear.	
10	SET the following objects: phaseRedClear.Table_Row.	Pass/Fail
11	GET the following objects: phaseRedClear.Table_Row.	Pass/Fail
12	VERIFY phaseRedClear.Table_Row IS_EQUAL_TO OriginalPhaseRedClear.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.8 Configure Phase Red Revert Time

Test Procedure:	**Configure Phase Red Revert Time**		
Description:	**This test case verifies that the ASC allows a management station to configure the minimum time a phase must display a RED indication before it may display a GREEN indication.**		
Requirement(s):	• **3.5.2.1.2.1.8 Configure Phase Red Revert Time**		
Variable(s):	**MaxRows** **OriginalPhaseRedRevert** **TestPhaseRedRevert** **Table_Row**	**maxPhases** **phaseRedRevert** **phaseRedRevert** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseRedRevert.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseRedRevert.	
4	ASSIGN TestPhaseRedRevert EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseRedRevert IS_EQUAL_TO OriginalPhaseRedRevert.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseRedRevert.Table_Row EQUALS TestPhaseRedRevert.	
6	SET the following objects: phaseRedRevert.Table_Row.	Pass/Fail
7	GET the following objects: phaseRedRevert.Table_Row.	Pass/Fail
8	VERIFY phaseRedRevert.Table_Row IS_EQUAL_TO TestPhaseRedRevert.	Pass/Fail
9	ASSIGN phaseRedRevert.Table_Row EQUALS OriginalPhaseRedRevert.	
10	SET the following objects: phaseRedRevert.Table_Row.	Pass/Fail
11	GET the following objects: phaseRedRevert.Table_Row.	Pass/Fail
12	VERIFY phaseRedRevert.Table_Row IS_EQUAL_TO OriginalPhaseRedRevert.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.9 Configure Unit Red Revert Time

	Configure Unit Red Revert Time

Test Procedure:			
Description:	**This test case verifies that the ASC allows a management station to configure the minimum red revert time for all phases.**		
Requirement(s):	• **3.5.2.1.2.1.9 Configure Unit Red Revert Time**		
Variable(s):	**OriginalRedRevert** **TestRedRevert**	**unitRedRevert** **unitRedRevert**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: unitRedRevert.	Pass/Fail
1.1	RECORD this information as OriginalRedRevert.	
2	ASSIGN TestRedRevert EQUALS RANDOM (0 TO 255).	
2.1	IF TestRedRevert IS_EQUAL_TO OriginalRedRevert.	
2.1.1	GOTO step 2.	
3	ASSIGN unitRedRevert EQUALS TestRedRevert.	
4	SET the following objects: unitRedRevert.	Pass/Fail
5	GET the following objects: unitRedRevert.	Pass/Fail
6	VERIFY unitRedRevert IS_EQUAL_TO TestRedRevert.	Pass/Fail
7	ASSIGN unitRedRevert EQUALS OriginalRedRevert.	
8	SET the following objects: unitRedRevert.	Pass/Fail
9	GET the following objects: unitRedRevert.	Pass/Fail
10	VERIFY unitRedRevert IS_EQUAL_TO OriginalRedRevert.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.10 Configure Phase Added Initial Time

Test Procedure:	**Configure Phase Added Initial Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the time by which the variable initial time period will be increased from zero with each vehicle actuation received during the associated vehicle clearance intervals.**	
Requirement(s):	• **3.5.2.1.2.1.10 Configure Phase Added Initial Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseAddedInitial** **TestPhaseAddedInitial**	**maxPhases** **Int** **phaseAddedInitial** **phaseAddedInitial**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseAddedInitial.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseAddedInitial.	
4	ASSIGN TestPhaseAddedInitial EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseAddedInitial IS_EQUAL_TO OriginalPhaseAddedInitial.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseAddedInitial.Table_Row EQUALS TestPhaseAddedInitial.	
6	SET the following objects: phaseAddedInitial.Table_Row.	Pass/Fail
7	GET the following objects: phaseAddedInitial.Table_Row.	Pass/Fail
8	VERIFY phaseAddedInitial.Table_Row IS_EQUAL_TO TestPhaseAddedInitial.	Pass/Fail
9	ASSIGN phaseAddedInitial.Table_Row EQUALS OriginalPhaseAddedInitial.	
10	SET the following objects: phaseAddedInitial.Table_Row.	Pass/Fail
11	GET the following objects: phaseAddedInitial.Table_Row.	Pass/Fail
12	VERIFY phaseAddedInitial.Table_Row IS_EQUAL_TO OriginalPhaseAddedInitial.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.11 Configure Phase Maximum Initial Time

Test Procedure:	**Configure Phase Maximum Initial Time**
Description:	**This test case verifies that the ASC allows a management station to configure the maximum a variable green time period of a phase can be increased.**
Requirement(s):	• **3.5.2.1.2.1.11 Configure Phase Maximum Initial Time**
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **OriginalPhaseMaximumInitial** **phaseMaximumInitial** **TestPhaseMaximumInitial** **phaseMaximumInitial**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseMaximumInitial.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseMaximumInitial.	
4	ASSIGN TestPhaseMaximumInitial EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseMaximumInitial IS_EQUAL_TO OriginalPhaseMaximumInitial.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseMaximumInitial.Table_Row EQUALS TestPhaseMaximumInitial.	
6	SET the following objects: phaseMaximumInitial.Table_Row.	Pass/Fail
7	GET the following objects: phaseMaximumInitial.Table_Row.	Pass/Fail
8	VERIFY phaseMaximumInitial.Table_Row IS_EQUAL_TO TestPhaseMaximumInitial.	Pass/Fail
9	ASSIGN phaseMaximumInitial.Table_Row EQUALS OriginalPhaseMaximumInitial.	
10	SET the following objects: phaseMaximumInitial.Table_Row.	Pass/Fail
11	GET the following objects: phaseMaximumInitial.Table_Row.	Pass/Fail
12	VERIFY phaseMaximumInitial.Table_Row IS_EQUAL_TO OriginalPhaseMaximumInitial.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.12 Configure Phase Time Before Reduction

Test Procedure:	**Configure Phase Time Before Reduction**
Description:	**This test case verifies that the ASC allows a management station to configure the time before the allowable gap between the Passage Time and Minimum Gap setting is reduced.**
Requirement(s):	• **3.5.2.1.2.1.12 Configure Phase Time Before Reduction**
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalTimeBeforeReduction** phaseTimeBeforeReduction **TestTimeBeforeReduction** phaseTimeBeforeReduction
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results

1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseTimeBeforeReduction.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTimeBeforeReduction.	
4	ASSIGN TestTimeBeforeReduction EQUALS RANDOM (0 TO 255).	
4.1	IF TestTimeBeforeReduction IS_EQUAL_TO OriginalTimeBeforeReduction.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseTimeBeforeReduction.Table_Row EQUALS TestTimeBeforeReduction.	
6	SET the following objects: phaseTimeBeforeReduction.Table_Row.	Pass/Fail
7	GET the following objects: phaseTimeBeforeReduction.Table_Row.	Pass/Fail
8	VERIFY phaseTimeBeforeReduction.Table_Row IS_EQUAL_TO TestTimeBeforeReduction.	Pass/Fail
9	ASSIGN phaseTimeBeforeReduction.Table_Row EQUALS OriginalTimeBeforeReduction.	
10	SET the following objects: phaseTimeBeforeReduction.Table_Row.	Pass/Fail
11	GET the following objects: phaseTimeBeforeReduction.Table_Row.	Pass/Fail
12	VERIFY phaseTimeBeforeReduction.Table_Row IS_EQUAL_TO OriginalTimeBeforeReduction.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.13 Configure Phase Time to Reduce

Test Procedure:	**Configure Phase Time to Reduce**	
Description:	**This test case verifies that the ASC allows a management station to configure the rate of reduction of the allowable gap between the Passage Time and Minimum Gap setting.**	
Requirement(s):	• **3.5.2.1.2.1.13 Configure Phase Time to Reduce**	
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalTimeToReduce** phaseTimeToReduce **TestTimeToReduce** phaseTimeToReduce	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail

1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseTimeToReduce.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTimeToReduce.	
4	ASSIGN TestTimeToReduce EQUALS RANDOM (0 TO 255).	
4.1	IF TestTimeToReduce IS_EQUAL_TO OriginalTimeToReduce.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseTimeToReduce.Table_Row EQUALS TestTimeToReduce.	
6	SET the following objects: phaseTimeToReduce.Table_Row.	Pass/Fail
7	GET the following objects: phaseTimeToReduce.Table_Row.	Pass/Fail
8	VERIFY phaseTimeToReduce.Table_Row IS_EQUAL_TO TestTimeToReduce.	Pass/Fail
9	ASSIGN phaseTimeToReduce.Table_Row EQUALS OriginalTimeToReduce.	
10	SET the following objects: phaseTimeToReduce.Table_Row.	Pass/Fail
11	GET the following objects: phaseTimeToReduce.Table_Row.	Pass/Fail
12	VERIFY phaseTimeToReduce.Table_Row IS_EQUAL_TO OriginalTimeToReduce.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.14 Configure Phase Cars Before Reduction

Test Procedure:	**Configure Phase Cars Before Reduction**	
Description:	**This test case verifies that the ASC allows a management station to configure the number of actuations on conflicting phases before the reduction of the allowable gap from the Passage Time shall begin.**	
Requirement(s):	• **3.5.2.1.2.1.14 Configure Phase Cars Before Reduction**	
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **OriginalCarsBeforeReduction** **phaseCarsBeforeReduction** **TestCarsBeforeReduction** **phaseCarsBeforeReduction**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	

3	GET the following objects: phaseCarsBeforeReduction.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalCarsBeforeReduction.	
4	ASSIGN Table_Row EQUALS RANDOM (0 TO 255).	
4.1	IF TestCarsBeforeReduction IS_EQUAL_TO OriginalCarsBeforeReduction.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseCarsBeforeReduction.Table_Row EQUALS TestCarsBeforeReduction.	
6	SET the following objects: phaseCarsBeforeReduction.Table_Row.	Pass/Fail
7	GET the following objects: phaseCarsBeforeReduction.Table_Row.	Pass/Fail
8	VERIFY phaseCarsBeforeReduction.Table_Row IS_EQUAL_TO TestCarsBeforeReduction.	Pass/Fail
9	ASSIGN phaseCarsBeforeReduction.Table_Row EQUALS OriginalCarsBeforeReduction.	
10	SET the following objects: phaseCarsBeforeReduction.Table_Row.	Pass/Fail
11	GET the following objects: phaseCarsBeforeReduction.Table_Row.	Pass/Fail
12	VERIFY phaseCarsBeforeReduction.Table_Row IS_EQUAL_TO OriginalCarsBeforeReduction.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.15 Configure Phase Reduce By Time

Test Procedure:	**Configure Phase Reduce By Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the rate of reduction for volume density gap reduction.**	
Requirement(s):	• **3.5.2.1.2.1.15 Configure Phase Reduce By Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseReduceBy** **TestPhaseReduceBy**	**maxPhases** **Int** **phaseReduceBy** **phaseReduceBy**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseReduceBy.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseReduceBy.	

4	ASSIGN TestPhaseReduceBy EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseReduceBy IS_EQUAL_TO OriginalPhaseReduceBy.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseReduceBy.Table_Row EQUALS TestPhaseReduceBy.	
6	SET the following objects: phaseReduceBy.Table_Row.	Pass/Fail
7	GET the following objects: phaseReduceBy.Table_Row.	Pass/Fail
8	VERIFY phaseReduceBy.Table_Row IS_EQUAL_TO TestPhaseReduceBy.	Pass/Fail
9	ASSIGN phaseReduceBy.Table_Row EQUALS OriginalPhaseReduceBy.	
10	SET the following objects: phaseReduceBy.Table_Row.	Pass/Fail
11	GET the following objects: phaseReduceBy.Table_Row.	Pass/Fail
12	VERIFY phaseReduceBy.Table_Row IS_EQUAL_TO OriginalPhaseReduceBy.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.16 Configure Phase Minimum Gap Time

Test Procedure:	**Configure Phase Minimum Gap Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the minimum time the gap between vehicles reported by detector actuations must be before a phase can be terminated.**	
Requirement(s):	• **3.5.2.1.2.1.16 Configure Phase Minimum Gap Time**	
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalPhaseMinimumGap** phaseMinimumGap **TestPhaseMinimumGap** phaseMinimumGap	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseMinimumGap.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseMinimumGap.	
4	ASSIGN TestPhaseMinimumGap EQUALS RANDOM (0 TO 255).	

4.1	IF TestPhaseMinimumGap IS_EQUAL_TO OriginalPhaseMinimumGap.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseMinimumGap.Table_Row EQUALS TestPhaseMinimumGap.	
6	SET the following objects: phaseMinimumGap.Table_Row.	Pass/Fail
7	GET the following objects: phaseMinimumGap.Table_Row.	Pass/Fail
8	VERIFY phaseMinimumGap.Table_Row IS_EQUAL_TO TestPhaseMinimumGap.	Pass/Fail
9	ASSIGN phaseMinimumGap.Table_Row EQUALS OriginalPhaseMinimumGap.	
10	SET the following objects: phaseMinimumGap.Table_Row.	Pass/Fail
11	GET the following objects: phaseMinimumGap.Table_Row.	Pass/Fail
12	VERIFY phaseMinimumGap.Table_Row IS_EQUAL_TO OriginalPhaseMinimumGap.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.17 Configure Phase Dynamic Maximum Limit

Test Procedure:	**Configure Phase Dynamic Maximum Limit**	
Description:	**This test case verifies that the ASC allows a management station to configure an upper or lower time limit of the GREEN Indication.**	
Requirement(s):	• **3.5.2.1.2.1.17 Configure Phase Dynamic Maximum Limit**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseDynamicMaxLimit** **TestPhaseDynamicMaxLimit**	**maxPhases** **Int** **phaseDynamicMaxLimit** **phaseDynamicMaxLimit**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseDynamicMaxLimit.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseDynamicMaxLimit.	
4	ASSIGN TestPhaseDynamicMaxLimit EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseDynamicMaxLimit IS_EQUAL_TO OriginalPhaseDynamicMaxLimit.	

4.1.1	GOTO step 4.	
5	ASSIGN phaseDynamicMaxLimit.Table_Row EQUALS TestPhaseDynamicMaxLimit.	
6	SET the following objects: phaseDynamicMaxLimit.Table_Row.	Pass/Fail
7	GET the following objects: phaseDynamicMaxLimit.Table_Row.	Pass/Fail
8	VERIFY phaseDynamicMaxLimit.Table_Row IS_EQUAL_TO TestPhaseDynamicMaxLimit.	Pass/Fail
9	ASSIGN phaseDynamicMaxLimit.Table_Row EQUALS OriginalPhaseDynamicMaxLimit.	
10	SET the following objects: phaseDynamicMaxLimit.Table_Row.	Pass/Fail
11	GET the following objects: phaseDynamicMaxLimit.Table_Row.	Pass/Fail
12	VERIFY phaseDynamicMaxLimit.Table_Row IS_EQUAL_TO OriginalPhaseDynamicMaxLimit.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.3.18 Configure Phase Dynamic Maximum Limit

Test Procedure:	**Configure Phase Dynamic Maximum Step**	
Description:	**This test case verifies that the ASC allows a management station to configure how much the allowable maximum time a phase's GREEN Indication may be incremented or decremented before reaching its limit.**	
Requirement(s):	• **3.5.2.1.2.1.18 Configure Phase Dynamic Maximum Step**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseDynamicMaxStep** **TestPhaseDynamicMaxStep**	**maxPhases** **Int** **phaseDynamicMaxStep** **phaseDynamicMaxStep**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseDynamicMaxStep.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseDynamicMaxStep.	
4	ASSIGN TestPhaseDynamicMaxStep EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseDynamicMaxStep IS_EQUAL_TO OriginalPhaseDynamicMaxStep.	
4.1.1	GOTO step 4.	

5	ASSIGN phaseDynamicMaxStep.Table_Row EQUALS TestPhaseDynamicMaxStep.	
6	SET the following objects: phaseDynamicMaxStep.Table_Row.	Pass/Fail
7	GET the following objects: phaseDynamicMaxStep.Table_Row.	Pass/Fail
8	VERIFY phaseDynamicMaxStep.Table_Row IS_EQUAL_TO TestPhaseDynamicMaxStep.	Pass/Fail
9	ASSIGN phaseDynamicMaxStep.Table_Row EQUALS OriginalPhaseDynamicMaxStep.	
10	SET the following objects: phaseDynamicMaxStep.Table_Row.	Pass/Fail
11	GET the following objects: phaseDynamicMaxStep.Table_Row.	Pass/Fail
12	VERIFY phaseDynamicMaxStep.Table_Row IS_EQUAL_TO OriginalPhaseDynamicMaxStep.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.19 Configure Phase Startup State

Test Procedure:	**Configure Phase Startup State**		
Description:	**This test case verifies that the ASC allows a management station to set a startup state for a phase.**		
Requirement(s):	• **3.5.2.1.2.1.19 Configure Phase Startup State**		
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseStartup** **TestPhaseStartup**	**maxPhases** **Int** **phaseStartup** **phaseStartup**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.		
2	GET the following objects: maxPhases.		Pass/Fail
2.1	RECORD this information as MaxRows.		
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).		
4	GET the following objects: phaseStartup.Table_Row.		Pass/Fail
4.1	RECORD this information as OriginalPhaseStartup.		
5	ASSIGN TestPhaseStartup EQUALS 2.		
5.1	NOTE 2 = phaseNotOn.		
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.		
7	ASSIGN phaseStartup.Table_Row EQUALS TestPhaseStartup.		
8	SET the following objects: phaseStartup.Table_Row.		Pass/Fail

9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseStartup.Table_Row.	Pass/Fail
11	VERIFY phaseStartup.Table_Row IS_EQUAL_TO TestPhaseStartup.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseStartup.Table_Row EQUALS OriginalPhaseStartup.	
14	SET the following objects: phaseStartup.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseStartup.Table_Row.	Pass/Fail
17	VERIFY phaseStartup.Table_Row IS_EQUAL_TO OriginalPhaseStartup.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.20 Configure Automatic Flash Entry Phase

Test Procedure:	**Configure Automatic Flash Entry Phase**		
Description:	**This test case verifies that the ASC allows a management station to configure which phases must be serviced before going into Automatic Flash.**		
Requirement(s):	• **3.5.2.1.2.1.20 Configure Automatic Flash Entry Phase**		
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 2.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail

9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.21 Configure Automatic Flash Exit Phase

Test Procedure:	**Configure Automatic Flash Exit Phase**		
Description:	**This test case verifies that the ASC allows a management station to configure which phases are serviced when Automatic Flash ends.**		
Requirement(s):	• **3.5.2.1.2.1.21 Configure Automatic Flash Exit Phase**		
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.		
2	GET the following objects: maxPhases.		Pass/Fail
2.1	RECORD this information as MaxRows.		
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).		
4	GET the following objects: phaseOptions.Table_Row.		Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.		
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 4.		
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.		
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.		
8	SET the following objects: phaseOptions.Table_Row.		Pass/Fail

9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.3.22 Configure Call to Non-Actuated 1

Test Procedure:	**Configure Call to Non-Actuated 1**		
Description:	**This test case verifies that the ASC allows a management station to configure a phase to respond to the Call to Non-Actuated 1 input.**		
Requirement(s):	• **3.5.2.1.2.1.22 Configure Call to Non-Actuated 1**		
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 8.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	

10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.23 Configure Call to Non-Actuated 2

Test Procedure:	**Configure Call to Non-Actuated 2**	
Description:	**This test case verifies that the ASC allows a management station to configure a phase to respond to the Call to Non-Actuated 2 input.**	
Requirement(s):	• **3.5.2.1.2.1.23 Configure Call to Non-Actuated 2**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 16.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail

11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.24 Configure Non-Lock Detector Memory

Test Procedure:	**Configure Non-Lock Detector Memory**
Description:	**This test case verifies that the ASC allows a management station to cause a call to the phase to lock at the beginning of the phase's yellow change interval. This allows calls to remain even if actuations are removed before the phase is serviced again.**
Requirement(s):	• **3.5.2.1.2.1.24 Configure Non-Lock Detector Memory** • **3.5.2.1.2.1.41 Configure Non-Lock Ped Detector Memory**
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **OriginalPhaseOptions** **phaseOptions** **TestPhaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 32.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	

10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.25 Configure Phase Minimum Vehicle Recall

Test Procedure:	**Configure Phase Minimum Recall**		
Description:	**This test case verifies that the ASC allows a management station to cause a recurring call for vehicle service exists for a phase to be served for at least the phase's minimum Green time.**		
Requirement(s):	• **3.5.2.1.2.1.25 Configure Phase Minimum Vehicle Recall**		
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 64.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	

10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.26 Configure Phase Maximum Vehicle Recall

Test Procedure:	**Configure Phase Maximum Vehicle Recall**	
Description:	**This test case verifies that the ASC allows a management station to cause a recurring call for a phase to be served for the maximum time that may be allocated to the phase.**	
Requirement(s):	• **3.5.2.1.2.1.26 Configure Phase Maximum Vehicle Recall**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	PERFORM the Test Procedure "A valid timing plan is loaded".	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS 'OriginalPhaseOptions XOR 128'.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'..	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail

11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.27 Configure Phase Soft Vehicle Recall

Test Procedure:	**Configure Phase Soft Vehicle Recall**	
Description:	**This test case verifies that the ASC allows a management station to cause a call is to be placed on a phase when all conflicting phases are in resting in green or red, and there are no serviceable conflicting calls.**	
Requirement(s):	• **3.5.2.1.2.1.27 Configure Phase Soft Vehicle Recall**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 512.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail

11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.28 Configure Dual Phase Entry

Test Procedure:	**Configure Dual Phase Entry**	
Description:	**This test case verifies that the ASC allows a management station to configure a phase to be active upon entry into a concurrency group when no calls exist in its own ring.**	
Requirement(s):	• **3.5.2.1.2.1.28 Configure Dual Phase Entry**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as MaxRows.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 1024.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail

12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.29 Configure Simultaneous Gap Disable

Test Procedure:	**Configure Simultaneous Gap Disable**	
Description:	**This test case verifies that the ASC allows a management station to configure if a gapped-out phase is allowed to revert to the extensible portion of the phase.**	
Requirement(s):	• **3.5.2.1.2.1.29 Configure Simultaneous Gap Disable**	
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalPhaseOptions** phaseOptions **TestPhaseOptions** phaseOptions	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as MaxRows.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 2048.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	

13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.30 Configure Guaranteed Passage

Test Procedure:	**Configure Guaranteed Passage**	
Description:	**This test case verifies that the ASC allows a management station to enable/disable volume density mode.**	
Requirement(s):	• **3.5.2.1.2.1.30 Configure Guaranteed Passage**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 4096.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	

14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.31 Configure Actuated Rest-in-Walk

Test Procedure:	**Configure Actuated Rest-in-Walk**	
Description:	**This test case verifies that the ASC allows a management station to set a phase to rest in Walk if there is no serviceable conflicting call at the end of the Walk time or if Maximum Vehicle Recall is enabled.**	
Requirement(s):	• **3.5.2.1.2.1.31 Configure Actuated Rest-in-Walk**	
Variable(s):	**MaxRows**	**maxPhases**
	Table_Row	**Int**
	OriginalPhaseOptions	**phaseOptions**
	TestPhaseOptions	**phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 8192.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	

14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.32 Configure Conditional Service Enable

Test Procedure:	**Configure Conditional Service Enable**	
Description:	**This test case verifies that the ASC allows a management station to enable/disable conditional service.**	
Requirement(s):	• **3.5.2.1.2.1.32 Configure Conditional Service Enable**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as MaxRows.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 16384.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	

14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.33 Configure Added Initial Calculation

Test Procedure:	**Configure Added Initial Calculation**	
Description:	**This test case verifies that the ASC allows a management station to set what detector values to use for the calculation of the variable portion of the green time.**	
Requirement(s):	• **3.5.2.1.2.1.33 Configure Added Initial Calculation**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOptions** **TestPhaseOptions**	**maxPhases** **Int** **phaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 32768.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	

14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.34 Configure Phase-to-Ring Association

Test Procedure:	**Configure Phase-to-Ring Association**	
Description:	**This test case verifies that the ASC allows a management station to set which ring a phase is in for a sequence.**	
Requirement(s):	• **3.5.2.1.2.1.34 Configure Phase-to-Ring Association**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseRing** **TestPhaseRing**	**maxPhases** **Int** **phaseRing** **phaseRing**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseRing.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseRing.	
5	USER-ACTION 'Select a valid value for phaseRing.Table_Row.'	
5.1	RECORD this information as TestPhaseRing.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseRing.Table_Row EQUALS TestPhaseRing.	
8	SET the following objects: phaseRing.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseRing.Table_Row.	Pass/Fail
11	VERIFY phaseRing.Table_Row IS_EQUAL_TO TestPhaseRing.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseRing.Table_Row EQUALS OriginalPhaseRing.	

14	SET the following objects: phaseRing.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseRing.Table_Row.	Pass/Fail
17	VERIFY phaseRing.Table_Row IS_EQUAL_TO OriginalPhaseRing.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.3.35 Configure Phase Concurrency

Test Procedure:	**Configure Phase Concurrency**	
Description:	**This test case verifies that the ASC allows a management station to configure what other phases that may run concurrently with a phase.**	
Requirement(s):	• **3.5.2.1.2.1.35 Configure Phase Concurrency**	
Variable(s):	**MaxRows** **Table_Row** **OriginalConcurrency** **TestConcurrency**	**maxPhases** **Int** **phaseConcurrency** **phaseConcurrency**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseConcurrency.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalConcurrency.	
5	USER-ACTION 'Select a valid value for phaseConcurrency.Table_Row. Each octet must not be greater than MaxRows .'	
5.1	RECORD this information as TestConcurrency.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseConcurrency.Table_Row EQUALS TestConcurrency.	
8	SET the following objects: phaseConcurrency.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseConcurrency.Table_Row.	Pass/Fail
11	VERIFY phaseConcurrency.Table_Row IS_EQUAL_TO TestConcurrency.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseConcurrency.Table_Row EQUALS OriginalConcurrency.	

14	SET the following objects: phaseConcurrency.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure "C.3.1.2 Verify Database Status and Verify Database Error".	
16	GET the following objects: phaseConcurrency.Table_Row.	Pass/Fail
17	VERIFY phaseConcurrency.Table_Row IS_EQUAL_TO OriginalConcurrency.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.36 Configure Pedestrian Clearance Time Allowed During Vehicle Clearance

Test Procedure:	**Configure Pedestrian Clearance Time Allowed During Vehicle Clearance**
Description:	**This test case verifies that the ASC allows a management station to configure how long a pedestrian clearance interval may extend into the yellow change and red clearance intervals.**
Requirement(s):	• **3.5.2.1.2.1.36 Configure Pedestrian Clearance Time Allowed During Vehicle Clearance**
Variable(s):	**MaxRows** **Table_Row** **OriginalTime** **TestTime** **maxPhases** **Int** **phasePedClearDuringVehicleClear** **phasePedClearDuringVehicleClear**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phasePedClearDuringVehicleClear.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTime.	
4	ASSIGN TestTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestTime IS_EQUAL_TO OriginalTime.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePedClearDuringVehicleClear.Table_Row EQUALS TestTime.	
6	SET the following objects: phasePedClearDuringVehicleClear.Table_Row.	Pass/Fail
7	GET the following objects: phasePedClearDuringVehicleClear.Table_Row.	Pass/Fail

8	VERIFY phasePedClearDuringVehicleClear.Table_Row IS_EQUAL_TO TestTime.	Pass/Fail
9	ASSIGN phasePedClearDuringVehicleClear.Table_Row EQUALS OriginalTime.	
10	SET the following objects: phasePedClearDuringVehicleClear.Table_Row.	Pass/Fail
11	GET the following objects: phasePedClearDuringVehicleClear.Table_Row.	Pass/Fail
12	VERIFY phasePedClearDuringVehicleClear.Table_Row IS_EQUAL_TO OriginalTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.37 Configure Pedestrian Walk Time

Test Procedure:	**Configure Pedestrian Walk Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the pedestrian walk time for a phase.**	
Requirement(s):	• **3.5.2.1.2.1.37 Configure Phase Walk Time**	
Variable(s):	**Table_Row** **Int** **OriginalPhaseWalk** **phaseWalk** **TestPhaseWalk** **phaseWalk** **MaxRows** **maxPhases**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseWalk.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseWalk.	
4	ASSIGN TestPhaseWalk EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseWalk IS_EQUAL_TO OriginalPhaseWalk.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseWalk.Table_Row EQUALS TestPhaseWalk.	
6	SET the following objects: phaseWalk.Table_Row.	Pass/Fail
7	GET the following objects: phaseWalk.Table_Row.	Pass/Fail
8	VERIFY phaseWalk.Table_Row IS_EQUAL_TO TestPhaseWalk.	Pass/Fail
9	ASSIGN phaseWalk.Table_Row EQUALS OriginalPhaseWalk.	

10	SET the following objects: phaseWalk.Table_Row.	Pass/Fail
11	GET the following objects: phaseWalk.Table_Row.	Pass/Fail
12	VERIFY phaseWalk.Table_Row IS_EQUAL_TO OriginalPhaseWalk.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.38 Configure Pedestrian Clearance Time

Test Procedure:	**Configure Pedestrian Clearance Time**		
Description:	**This test case verifies that the ASC allows a management station the pedestrian clearance time for a phase.**		
Requirement(s):	• **3.5.2.1.2.1.38 Configure Pedestrian Clearance Time**		
Variable(s):	**MaxRows** **OriginalPhasePedClear** **TestPhasePedClear** **Table_Row**	**maxPhases** **phasePedestrianClear** **phasePedestrianClear** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phasePedestrianClear.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhasePedClear.	
4	ASSIGN TestPhasePedClear EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhasePedClear IS_EQUAL_TO OriginalPhasePedClear.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePedestrianClear.Table_Row EQUALS TestPhasePedClear.	
6	SET the following objects: phasePedestrianClear.Table_Row.	Pass/Fail
7	GET the following objects: phasePedestrianClear.Table_Row.	Pass/Fail
8	VERIFY phasePedestrianClear.Table_Row IS_EQUAL_TO TestPhasePedClear.	Pass/Fail
9	ASSIGN phasePedestrianClear.Table_Row EQUALS OriginalPhasePedClear.	
10	SET the following objects: phasePedestrianClear.Table_Row.	Pass/Fail
11	GET the following objects: phasePedestrianClear.Table_Row.	Pass/Fail
12	VERIFY phasePedestrianClear.Table_Row IS_EQUAL_TO OriginalPhasePedClear.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.39 Configure Pedestrian Phase Walk Service Limit

Test Procedure:	**Configure Pedestrian Phase Walk Service Limit**
Description:	**This test case verifies that the ASC allows a management station to configure how many times the pedestrian Walk indication to be shown again within the same cycle.**
Requirement(s):	• **3.5.2.1.2.1.39 Configure Pedestrian Phase Walk Service Limit**
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalLimit** phasePedServiceLimit **TestPedLimit** phasePedServiceLimit
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phasePedServiceLimit.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalLimit	
4	ASSIGN TestLimit EQUALS RANDOM (0 TO 255).	
4.1	IF TestLimit IS_EQUAL_TO OriginalLimit.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePedServiceLimit.Table_Row EQUALS TestLimit.	
6	SET the following objects: phasePedServiceLimit.Table_Row.	Pass/Fail
7	GET the following objects: phasePedServiceLimit.Table_Row.	Pass/Fail
8	VERIFY phasePedServiceLimit.Table_Row IS_EQUAL_TO TestLimit.	Pass/Fail
9	ASSIGN phasePedServiceLimit.Table_Row EQUALS OriginalLimit.	
10	SET the following objects: phasePedServiceLimit.Table_Row.	Pass/Fail
11	GET the following objects: phasePedServiceLimit.Table_Row.	Pass/Fail
12	VERIFY phasePedServiceLimit.Table_Row IS_EQUAL_TO OriginalLimit.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.40 Configure Pedestrian Phase Don't Walk Revert Time

Test Procedure:	**Configure Pedestrian Phase Don't Walk Revert Time**
Description:	**This test case verifies that the ASC allows a management station to configure the minimum time for a Don't Walk indication.**
Requirement(s):	• **3.5.2.1.2.1.40 Configure Pedestrian Phase Don't Walk Revert Time**
Variable(s):	**MaxRows** maxPhases **Table_Row** Int **OriginalDontWalkRevert** phaseDontWalkRevert **TestPedDontWalkRevert** phaseDontWalkRevert
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phaseDontWalkRevert.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDontWalkRevert.	
4	ASSIGN TestDontWalkRevert EQUALS RANDOM (0 TO 255).	
4.1	IF TestDontWalkRevert IS_EQUAL_TO OriginalDontWalkRevert.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseDontWalkRevert.Table_Row EQUALS TestDontWalkRevert.	
6	SET the following objects: phaseDontWalkRevert.Table_Row.	Pass/Fail
7	GET the following objects: phaseDontWalkRevert.Table_Row.	Pass/Fail
8	VERIFY phaseDontWalkRevert.Table_Row IS_EQUAL_TO TestDontWalkRevert.	Pass/Fail
9	ASSIGN phaseDontWalkRevert.Table_Row EQUALS OriginalDontWalkRevert.	
10	SET the following objects: phaseDontWalkRevert.Table_Row.	Pass/Fail
11	GET the following objects: phaseDontWalkRevert.Table_Row.	Pass/Fail
12	VERIFY phaseDontWalkRevert.Table_Row IS_EQUAL_TO OriginalDontWalkRevert.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.41 Configure Pedestrian Phase Recall

	Configure Pedestrian Phase Recall

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to cause 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.**
Requirement(s):	• **3.5.2.1.2.1.42 Configure Pedestrian Phase Recall**
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **OriginalPhaseOptions** **phaseOptions** **TestPhaseOptions** **phaseOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhaseOptions.	
5	ASSIGN TestPhaseOptions EQUALS OriginalPhaseOptions XOR 256.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN phaseOptions.Table_Row EQUALS TestPhaseOptions.	
8	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
11	VERIFY phaseOptions.Table_Row IS_EQUAL_TO TestPhaseOptions.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN phaseOptions.Table_Row EQUALS OriginalPhaseOptions.	
14	SET the following objects: phaseOptions.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: phaseOptions.Table_Row.	Pass/Fail
17	VERIFY phaseOptions.Table_Row IS_EQUAL_TO OriginalPhaseOptions.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.42 Configure Phase Alternate Pedestrian Clearance Time

Test Procedure:	**Configure Phase Alternate Pedestrian Clearance Time**
Description:	**This test case verifies that the ASC allows a management station to configure an alternate pedestrian clearance time for a phase.**
Requirement(s):	• **3.5.2.1.2.1.43 Configure Alternate Pedestrian Clearance Time** • **3.5.2.1.13.1.3 Configure APS Extra Crossing Time**
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **OriginalPedAlternateClearance** **phasePedAlternateClearance** **TestPedAlternateClearance** **phasePedAlternateClearance**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phasePedAlternateClearance.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPedAlternateClearance.	
4	ASSIGN TestPedAlternateClearance EQUALS RANDOM (0 TO 255).	
4.1	IF TestPedAlternateClearance IS_EQUAL_TO OriginalPedAlternateClearance.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePedAlternateClearance.Table_Row EQUALS TestPedAlternateClearance.	
6	SET the following objects: phasePedAlternateClearance.Table_Row.	Pass/Fail
7	GET the following objects: phasePedAlternateClearance.Table_Row.	Pass/Fail
8	VERIFY phasePedAlternateClearance.Table_Row IS_EQUAL_TO TestPedAlternateClearance.	Pass/Fail
9	ASSIGN phasePedAlternateClearance.Table_Row EQUALS OriginalPedAlternateClearance.	
10	SET the following objects: phasePedAlternateClearance.Table_Row.	Pass/Fail
11	GET the following objects: phasePedAlternateClearance.Table_Row.	Pass/Fail
12	VERIFY phasePedAlternateClearance.Table_Row IS_EQUAL_TO OriginalPedAlternateClearance.	Pass/Fail

Test Procedure Results

Tested By:	**Date Tested:**	**Pass/Fail**

Test Procedure Notes:

C.3.3.43 Configure Phase Alternate Pedestrian Walk Time

	Configure Phase Alternate Pedestrian Walk Time

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to configure an alternate pedestrian walk time for a phase.**
Requirement(s):	• **3.5.2.1.2.1.44 Configure Alternate Pedestrian Walk Time** • **3.5.2.1.13.1.3 Configure APS Extra Crossing Time**
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **OriginalPedAlternateWalk** **phasePedAlternateWalk** **TestPedAlternateWalk** **phasePedAlternateWalk**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phasePedAlternateWalk.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPedAlternateWalk.	
4	ASSIGN TestPedAlternateWalk EQUALS RANDOM (0 TO 255).	
4.1	IF TestPedAlternateWalk IS_EQUAL_TO OriginalPedAlternateWalk.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePedAlternateWalk.Table_Row EQUALS TestPedAlternateWalk.	
6	SET the following objects: phasePedAlternateWalk.Table_Row.	Pass/Fail
7	GET the following objects: phasePedAlternateWalk.Table_Row.	Pass/Fail
8	VERIFY phasePedAlternateWalk.Table_Row IS_EQUAL_TO TestPedAlternateWalk.	Pass/Fail
9	ASSIGN phasePedAlternateWalk.Table_Row EQUALS OriginalPedAlternateWalk.	
10	SET the following objects: phasePedAlternateWalk.Table_Row.	Pass/Fail
11	GET the following objects: phasePedAlternateWalk.Table_Row.	Pass/Fail
12	VERIFY phasePedAlternateWalk.Table_Row IS_EQUAL_TO OriginalPedAlternateWalk.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.3.44 Configure Pedestrian Phase Advanced Walk Time

Test Procedure:	**Configure Pedestrian Phase Advanced Walk Time**

Description:	**This test case verifies that the ASC allows a management station to configure the pedestrian WALK indication to be displayed before the start of the GREEN indication**
Requirement(s):	• **3.5.2.1.2.1.45 Configure Pedestrian Phase Advanced Walk Time**
Variable(s):	**MaxRows** **maxPhases** **Table_Row** **Int** **OriginalAdvanceTime** **phasePedAdvanceDelayTime** **TestAdvanceTime** **phasePedAdvanceDelayTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phasePedAdvanceWalkTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalAdvanceTime.	
4	ASSIGN TestAdvanceTimeTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestAdvanceTime IS_EQUAL_TO OriginalAdvanceTime.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePedAdvanceWalkTime.Table_Row EQUALS TestAdvanceTime.	
6	SET the following objects: phasePedAdvanceWalkTime.Table_Row.	Pass/Fail
7	GET the following objects: phasePedAdvanceWalkTime.Table_Row.	Pass/Fail
8	VERIFY phasePedAdvanceWalkTime.Table_Row IS_EQUAL_TO TestAdvanceTime.	Pass/Fail
9	ASSIGN phasePedAdvanceWalkTime.Table_Row EQUALS OriginalAdvanceTime.	
10	SET the following objects: phasePedAdvanceWalkTime.Table_Row.	Pass/Fail
11	GET the following objects: phasePedAdvanceWalkTime.Table_Row.	Pass/Fail
12	VERIFY phasePedAdvanceWalkTime.Table_Row IS_EQUAL_TO OriginalAdvanceTime.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.45 Configure Pedestrian Phae Delayed Walk Time

Test Procedure:	**Configure Pedestrian Phase Delayed Walk Time**
Description:	**This test case verifies that the ASC allows a management station to configure the pedestrian WALK indication to be displayed after the start of the GREEN indication**

Requirement(s):	• **3.5.2.1.2.1.46 Configure Pedestrian Phase Delayed Walk Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDelayTime** **TestDelayTime**	**maxPhases** **Int** **phasePedDelayTime** **phasePedDelayTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phasePedDelayTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDelayTime.	
4	ASSIGN TestDelayTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestDelayTime IS_EQUAL_TO OriginalDelayTime.	
4.1.1	GOTO step 4.	
5	ASSIGN phasePedDelayTime.Table_Row EQUALS TestDelayTime.	
6	SET the following objects: phasePedDelayTime.Table_Row.	Pass/Fail
7	GET the following objects: phasePedDelayTime.Table_Row.	Pass/Fail
8	VERIFY phasePedDelayTime.Table_Row IS_EQUAL_TO TestDelayTime.	Pass/Fail
9	ASSIGN phasePedDelayTime.Table_Row EQUALS OriginalDelayTime.	
10	SET the following objects: phasePedDelayTime.Table_Row.	Pass/Fail
11	GET the following objects: phasePedDelayTime.Table_Row.	Pass/Fail
12	VERIFY phasePedDelayTime.Table_Row IS_EQUAL_TO OriginalDelayTime.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.46 Configure Advance Warning Green

Test Procedure:	**Configure Advance Green Warning**	
Description:	**This test case verifies that the ASC allows a management station to configure the time for an advanced warning signal before a GREEN indication.**	
Requirement(s):	• **3.5.2.1.2.1.47 Configure Advance Green Warning**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTime**	**maxPhases** **Int** **phaseAdvWarnGrnStartTime**

	TestTime	**phaseAdvWarnGrnStartTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phaseAdvWarnGrnStartTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTime.	
4	ASSIGN TestTime EQUALS RANDOM (0 TO 255).	
4.1	NOTE 'The value should not exceed the total amount of the clearance time of the phase that is being terminated prior to the start of this phase'.	
4.1	IF TestTime IS_EQUAL_TO OriginalTime.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseAdvWarnGrnStartTime.Table_Row EQUALS TestTime.	
6	SET the following objects: phaseAdvWarnGrnStartTime.Table_Row.	Pass/Fail
7	GET the following objects: phaseAdvWarnGrnStartTime.Table_Row.	Pass/Fail
8	VERIFY phaseAdvWarnGrnStartTime.Table_Row IS_EQUAL_TO TestTime.	Pass/Fail
9	ASSIGN phaseAdvWarnGrnStartTime.Table_Row EQUALS OriginalTime.	
10	SET the following objects: phaseAdvWarnGrnStartTime.Table_Row.	Pass/Fail
11	GET the following objects: phaseAdvWarnGrnStartTime.Table_Row.	Pass/Fail
12	VERIFY phaseAdvWarnGrnStartTime.Table_Row IS_EQUAL_TO OriginalTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.47 Configure Advance Warning Red

Test Procedure:	**Configure Advance Red Warning**	
Description:	**This test case verifies that the ASC allows a management station to configure the time for an advanced warning signal before a RED indication.**	
Requirement(s):	• **3.5.2.1.2.1.48 Configure Advance Red Warning**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTime** **TestTime**	**maxPhases** **Int** **phaseAdvWarnRedStartTime** **phaseAdvWarnRedStartTime**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phaseAdvWarnRedStartTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTime.	
4	ASSIGN TestTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestTime IS_EQUAL_TO OriginalTime.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseAdvWarnRedStartTime.Table_Row EQUALS TestTime.	
6	SET the following objects: phaseAdvWarnRedStartTime.Table_Row.	Pass/Fail
7	GET the following objects: phaseAdvWarnRedStartTime.Table_Row.	Pass/Fail
8	VERIFY phaseAdvWarnRedStartTime.Table_Row IS_EQUAL_TO TestTime.	Pass/Fail
9	ASSIGN phaseAdvWarnRedStartTime.Table_Row EQUALS OriginalTime.	
10	SET the following objects: phaseAdvWarnRedStartTime.Table_Row.	Pass/Fail
11	GET the following objects: phaseAdvWarnRedStartTime.Table_Row.	Pass/Fail
12	VERIFY phaseAdvWarnRedStartTime.Table_Row IS_EQUAL_TO OriginalTime.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.48 Configure Alternate Minimum Green During Transition

Test Procedure:	**Configure Alternate Minimum Green During Transition**
Description:	**This test case verifies that the ASC allows a management station to configure an alternate minimum green time to be used during transition between cycles.**
Requirement(s):	• **3.5.2.1.2.1.51 Configure Alternate Minimum Vehicle Green Time during Transition**
Variable(s):	**MaxRows** — **maxPhases** **Table_Row** — **Int** **OriginalTime** — **phaseAltMinTimeTransition** **TestTime** — **phaseAltMinTimeTransition**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results

1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phaseAltMinTimeTransition.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTime.	
4	ASSIGN TestTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestTime IS_EQUAL_TO OriginalTime.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseAltMinTimeTransition.Table_Row EQUALS TestTime.	
6	SET the following objects: phaseAltMinTimeTransition.Table_Row.	Pass/Fail
7	GET the following objects: phaseAltMinTimeTransition.Table_Row.	Pass/Fail
8	VERIFY phaseAltMinTimeTransition.Table_Row IS_EQUAL_TO TestTime.	Pass/Fail
9	ASSIGN phaseAltMinTimeTransition.Table_Row EQUALS OriginalTime.	
10	SET the following objects: phaseAltMinTimeTransition.Table_Row.	Pass/Fail
11	GET the following objects: phaseAltMinTimeTransition.Table_Row.	Pass/Fail
12	VERIFY phaseAltMinTimeTransition.Table_Row IS_EQUAL_TO OriginalTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.49 Configure Alternate Pedestrian Walk Time During Transition

Test Procedure:	**Configure Alternate Pedestrian Walk Time During Transition**	
Description:	**This test case verifies that the ASC allows a management station to configure an alternate minimum walk time to be used during transition between cycles.**	
Requirement(s):	• **3.5.2.1.2.1.52 Configure Alternate Minimum Pedestrian Walk Time during Transition**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTime** **TestTime**	**maxPhases** **Int** **phaseWalkDuringTransition** **phaseWalkDuringTransition**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phaseWalkDuringTransition.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTime.	

4	ASSIGN TestTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestTime IS_EQUAL_TO OriginalTime.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseWalkDuringTransition.Table_Row EQUALS TestTime.	
6	SET the following objects: phaseWalkDuringTransition.Table_Row.	Pass/Fail
7	GET the following objects: phaseWalkDuringTransition.Table_Row.	Pass/Fail
8	VERIFY phaseWalkDuringTransition.Table_Row IS_EQUAL_TO TestTime.	Pass/Fail
9	ASSIGN phaseWalkDuringTransition.Table_Row EQUALS OriginalTime.	
10	SET the following objects: phaseWalkDuringTransition.Table_Row.	Pass/Fail
11	GET the following objects: phaseWalkDuringTransition.Table_Row.	Pass/Fail
12	VERIFY phaseWalkDuringTransition.Table_Row IS_EQUAL_TO OriginalTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.50 Configure Alternate Pedestrian Clearance Time During Transition

Test Procedure:	**Configure Alternate Pedestrian Clearance Time During Transition**	
Description:	**This test case verifies that the ASC allows a management station to configure an alternate minimum pedestrian clearance time to be used during transition between cycles.**	
Requirement(s):	• **3.5.2.1.2.1.53 Configure Alternate Minimum Pedestrian Clearance Time during Transition**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTime** **TestTime**	**maxPhases** **Int** **phasePedClearDuringTransition** **phasePedClearDuringTransition**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhases.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: phasPedClearDuringTransition.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTime.	
4	ASSIGN TestTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestTime IS_EQUAL_TO OriginalTime.	

4.1.1	GOTO step 4.	
5	ASSIGN phasePedClearDuringTransition.Table_Row EQUALS TestTime.	
6	SET the following objects: phasePedClearDuringTransition.Table_Row.	Pass/Fail
7	GET the following objects: phasePedClearDuringTransition.Table_Row.	Pass/Fail
8	VERIFY phasePedClearDuringTransition.Table_Row IS_EQUAL_TO TestTime.	Pass/Fail
9	ASSIGN phasePedClearDuringTransition.Table_Row EQUALS OriginalTime.	
10	SET the following objects: phasePedClearDuringTransition.Table_Row.	Pass/Fail
11	GET the following objects: phasePedClearDuringTransition.Table_Row.	Pass/Fail
12	VERIFY phasePedClearDuringTransition.Table_Row IS_EQUAL_TO OriginalTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.51 Determine Maximum Number of Phases

Test Procedure:	**Determine Maximum Number of Phases**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of phases supported by ASC and verifies that the ASC supports the minimum number of phases required by the user.**	
Requirement(s):	• **3.4.1.1 Retrieve Data** • **3.5.2.1.2.2.1 Determine Maximum Number of Phases**	
Variable(s):	**UserMinPhases** **maxPhases**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of phases required as specified in FR ID 3.5.2.1.2.3.1 of the PRL.'	
1.1	RECORD this information as UserMinPhases.	
2	GET the following objects: maxPhases.	Pass/Fail
3	VERIFY maxPhases IS_NOT_LESS_THAN UserMinPhases.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.3.52 Determine Maximum Number of Phase Sets

Test Procedure:	**Determine Maximum Number of Phases Sets**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of phase sets supported by ASC and verifies that the ASC supports the minimum number of phase sets required by the user.**
Requirement(s):	• **3.5.2.1.2.2.2 Determine Maximum Number of Phase Sets**
Variable(s):	**UserMinPhaseSets** **maxPhasesSets**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	USER-ACTION 'Determine the number of phases sets required as specified in FR ID 3.5.2.1.2.3.2 of the PRL.'	
1.1	RECORD this information as UserMinPhaseSets.	
2	GET the following objects: maxPhasesSets.	Pass/Fail
3	VERIFY maxPhases IS_NOT_LESS_THAN UserMinPhaseSets.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4 Coordination

C.3.4.1 Configure Operational Mode for Coordination

Test Procedure:	**Configure Operational Mode for Coordination**
Description:	**This test case verifies that the ASC allows a management station to the mode for operational mode for coordination.**
Requirement(s):	• **3.5.2.1.3.1 Configure Operational Mode for Coordination**
Variable(s):	**OriginalMode coordOperationalMode** **TestMode coordOperationalMode**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: coordOperationalMode.	Pass/Fail
1.1	RECORD this information as OriginalMode.	
2	ASSIGN TestMode EQUALS RANDOM (0 TO 255).	
2.1	IF TestMode IS_EQUAL_TO OriginalMode.	
2.1.1	GOTO step 2.	
3	ASSIGN coordOperationalMode EQUALS TestMode.	
4	SET the following objects: coordOperationalMode.	Pass/Fail
5	GET the following objects: coordOperationalMode.	Pass/Fail
6	VERIFY coordOperationalMode IS_EQUAL_TO TestMode.	Pass/Fail
7	ASSIGN coordOperationalMode EQUALS OriginalMode.	
8	SET the following objects: coordOperationalMode.	Pass/Fail
9	GET the following objects: coordOperationalMode.	Pass/Fail
10	VERIFY coordOperationalMode IS_EQUAL_TO OriginalMode.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.2 Configure Correction Mode for Coordination

Test Procedure:	**Configure Correction Mode for Coordination**
Description:	**This test case verifies that the ASC allows a management station to set the coordination correction mode.**
Requirement(s):	• **3.5.2.1.3.2 Configure Correction Mode for Coordination**
Variable(s):	**OriginalMode coordCorrectionMode** **TestMode coordCorrectionMode**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: coordCorrectionMode.	Pass/Fail
1.1	RECORD this information as OriginalMode.	
2	ASSIGN TestMode EQUALS RANDOM (2 TO 5).	
2.1	IF TestMode IS_EQUAL_TO OriginalMode.	
2.1.1	GOTO step 2.	
3	ASSIGN coordCorrectionMode EQUALS TestMode.	
4	SET the following objects: coordCorrectionMode.	Pass/Fail
5	GET the following objects: coordCorrectionMode.	Pass/Fail
6	VERIFY coordCorrectionMode IS_EQUAL_TO TestMode.	Pass/Fail
7	ASSIGN coordCorrectionMode EQUALS OriginalMode.	
8	SET the following objects: coordCorrectionMode.	Pass/Fail
9	GET the following objects: coordCorrectionMode.	Pass/Fail
10	VERIFY coordCorrectionMode IS_EQUAL_TO OriginalMode.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.3 Configure Maximum Mode for Coordination

Test Procedure:	Configure Maximum Mode for Coordination	
Description:	**This test case verifies that the ASC allow a management station to set the coordination maximum.**	
Requirement(s):	• **3.5.2.1.3.3 Configure Maximum Mode for Coordination**	
Variable(s):	**OriginalMode coordMaximumMode** **TestMode coordMaximumMode**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: coordMaximumMode.	Pass/Fail
1.1	RECORD this information as OriginalMode.	
2	ASSIGN TestMode EQUALS RANDOM (2 TO 5).	
2.1	IF TestMode IS_EQUAL_TO OriginalMode.	
2.1.1	GOTO step 2.	
3	ASSIGN coordMaximumMode EQUALS TestMode.	
4	SET the following objects: coordMaximumMode.	Pass/Fail
5	GET the following objects: coordMaximumMode.	Pass/Fail
6	VERIFY coordMaximumMode IS_EQUAL_TO TestMode.	Pass/Fail
7	ASSIGN coordMaximumMode EQUALS OriginalMode.	
8	SET the following objects: coordMaximumMode.	Pass/Fail
9	GET the following objects: coordMaximumMode.	Pass/Fail

10	VERIFY coordMaximumMode IS_EQUAL_TO OriginalMode.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.4 Configure Unit-Level Force Mode for Coordination

Test Procedure:	**Configure Unit-Level Force Mode for Coordination**	
Description:	**This test case verifies that the ASC allows a management station to set a default setting for forcing of phases during coordination.**	
Requirement(s):	• **3.5.2.1.3.4 Configure Unit-level Force Mode for Coordination**	
Variable(s):	**OriginalMode unitCoordForceMode** **TestMode unitCoordForceMode**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: unitCoordForceMode.	Pass/Fail
1.1	RECORD this information as OriginalMode.	
2	IF OriginalMode IS_EQUAL_TO 2.	
2.1	ASSIGN TestMode EQUALS 3.	
2.2	GOTO step 4	
3	ASSIGN TestMode EQUALS 2.	
4	ASSIGN coordForceMode EQUALS TestMode.	
5	SET the following objects: unitCoordForceMode.	Pass/Fail
6	GET the following objects: unitCoordForceMode.	Pass/Fail
7	VERIFY coordForceMode IS_EQUAL_TO TestMode.	Pass/Fail
8	ASSIGN coordForceMode EQUALS OriginalMode.	
9	SET the following objects: unitCoordForceMode.	Pass/Fail
10	GET the following objects: unitCoordForceMode.	Pass/Fail
11	VERIFY coordForceMode IS_EQUAL_TO OriginalMode.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.5 Configure Phase-Level Force Mode for Coordination

Test Procedure:	**Configure Phase-Level Force Mode for Coordination**
Description:	**This test case verifies that the ASC allows a management station to set the setting for forcing of phases during coordination that can override the default setting and vary by pattern.**

Requirement(s):	• **3.5.2.1.3.5 Configure Phase-level Force Mode for Coordination**
Variable(s):	**OriginalMode** **splitCoordForceMode** **TestMode** **splitCoordForceMode** **MaxSplits** **maxSplits** **MaxPhases** **maxPhases** **Selected_Phase** **Int** **Selected_Split** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSplits.	Pass/Fail
1.1	RECORD this information as MaxSplits.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxPhases.	
3	ASSIGN Selected_Split EQUALS RANDOM (1 TO MaxSplits).	
4	ASSIGN Selected_Phase EQUALS RANDOM (1 TO MaxPhases).	
5	GET the following objects: splitCoordForceMode.Selected_Split.Selected_Phase.	Pass/Fail
5.1	RECORD this information as OriginalForceMode.	
6	ASSIGN TestMode EQUALS RANDOM (2 TO 4).	
6.1	IF TestMode IS_EQUAL_TO OriginalMode.	
6.1.1	GOTO step 6.	
7	ASSIGN splitCoordForceMode.Selected_Split.Selected_Phase EQUALS TestMode.	
8	SET the following objects: splitCoordForceMode.Selected_Split.Selected_Phase.	Pass/Fail
9	GET the following objects: splitCoordForceMode.Selected_Split.Selected_Phase.	Pass/Fail
10	VERIFY splitCoordForceMode.Selected_Split.Selected_Phase IS_EQUAL_TO TestMode.	Pass/Fail
11	ASSIGN splitCoordForceMode.Selected_Split.Selected_Phase EQUALS OriginalMode.	
12	SET the following objects: splitCoordForceMode.Selected_Split.Selected_Phase.	Pass/Fail
13	GET the following objects: splitCoordForceMode.Selected_Split.Selected_Phase.	Pass/Fail
14	VERIFY splitCoordForceMode.Selected_Split.Selected_Phase IS_EQUAL_TO OriginalMode.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.6 Configure Pattern Reference Phase

	Configure Pattern Reference Phase

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to configure the phase that is to be active when the pattern cycle starts and ends.**
Requirement(s):	• **3.5.2.1.3.6 Configure Pattern Reference Phase**
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalPhase** **patternReferencePhase** **TestPhase** **patternReferencePhase**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: patternReferencePhase.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhase.	
4	ASSIGN TestPhase EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhase IS_EQUAL_TO OriginalPhase.	
4.1.1	GOTO step 4.	
5	ASSIGN patternReferencePhase.Table_Row EQUALS TestPhase.	
6	SET the following objects: patternReferencePhase.Table_Row.	Pass/Fail
7	GET the following objects: patternReferencePhase.Table_Row.	Pass/Fail
8	VERIFY patternReferencePhase.Table_Row IS_EQUAL_TO TestPhase.	Pass/Fail
9	ASSIGN patternReferencePhase.Table_Row EQUALS OriginalPhase.	
10	SET the following objects: patternReferencePhase.Table_Row.	Pass/Fail
11	GET the following objects: patternReferencePhase.Table_Row.	Pass/Fail
12	VERIFY patternReferencePhase.Table_Row IS_EQUAL_TO OriginalPhase.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.7 Configure Pattern Reference Point

Test Procedure:	**Configure Pattern Reference Point**
Description:	**This test case verifies that the ASC allows a management station to select an interval within the reference phase to specify the point when the pattern cycle starts and ends.**

Requirement(s):	• **3.5.2.1.3.7 Configure Pattern Reference Point**
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalPoint** **patternReferencePoint** **TestPoint** **patternReferencePoint**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: patternReferencePoint.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPoint.	
4	ASSIGN TestPoint EQUALS RANDOM (2 TO 5).	
4.1	IF TestPoint IS_EQUAL_TO OriginalPoint.	
4.1.1	GOTO step 4.	
5	ASSIGN patternReferencePhase.Table_Row EQUALS TestPoint.	
6	SET the following objects: patternReferencePoint.Table_Row.	Pass/Fail
7	GET the following objects: patternReferencePoint.Table_Row.	Pass/Fail
8	VERIFY patternReferencePoint.Table_Row IS_EQUAL_TO TestPoint.	Pass/Fail
9	ASSIGN patternReferencePoint.Table_Row EQUALS OriginalPoint.	
10	SET the following objects: patternReferencePoint.Table_Row.	Pass/Fail
11	GET the following objects: patternReferencePoint.Table_Row.	Pass/Fail
12	VERIFY patternReferencePoint.Table_Row IS_EQUAL_TO OriginalPoint.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.8 Configure Omit Phases During Transitions

Test Procedure:	**Configure Omit Phases During Transitions**
Description:	**This test case verifies that the ASC allows a management station to configure phases that may be omitted during a transition between cycles.**
Requirement(s):	• **3.5.2.1.3.8 Configure Omit Phases During Transitions**
Variable(s):	**OriginalSplitOptions** **splitOptions** **TestSplitOptions** **splitOptions** **MaxSplits** **maxSplits** **MaxPhases** **maxPhases**

	Selected_Phase **Int** **Selected_Split** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSplits.	Pass/Fail
1.1	RECORD this information as MaxSplits.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxPhases.	
3	ASSIGN Selected_Split EQUALS RANDOM (1 TO MaxSplits).	
4	ASSIGN Selected_Phase EQUALS RANDOM (1 TO MaxPhases).	
5	GET the following objects: splitOptions.Selected_Split.Selected_Phase.	Pass/Fail
5.1	RECORD this information as OriginalSplitOptions.	
6	ASSIGN TestSplitTime EQUALS OriginalSplitOptions XOR 1	
8	ASSIGN splitOptions.Selected_Split.Selected_Phase EQUALS TestSplitOptions.	
9	SET the following objects: splitOptions.Selected_Split.Selected_Phase.	Pass/Fail
10	GET the following objects: splitOptions.Selected_Split.Selected_Phase.	Pass/Fail
11	VERIFY splitOptions.Selected_Split.Selected_Phase IS_EQUAL_TO TestSplitOptions.	Pass/Fail
12	ASSIGN splitOptions.Selected_Split.Selected_Phase EQUALS OriginalSplitOptions.	
13	SET the following objects: splitOptions.Selected_Split.Selected_Phase.	Pass/Fail
14	GET the following objects: splitOptions.Selected_Split.Selected_Phase.	Pass/Fail
15	VERIFY splitOptions.Selected_Split.Selected_Phase IS_EQUAL_TO OriginalSplitOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.9 Configure Pattern Synchronization Time

Test Procedure:	**Configure Pattern Synchronization Time**
Description:	**This test case verifies that the ASC allows a management station to configure the time pattern synchronization time for when the master cycle begins.**
Requirement(s):	• **3.5.2.1.3.9 Configure Pattern Synchronization Time**
Variable(s):	**OriginalPatternSync** **timebaseAscPatternSync** **TestPatternSync** **timebaseAscPatternSync**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: timebaseAscPatternSync.	Pass/Fail
1.1	RECORD this information as OriginalPatternSync.	
2	ASSIGN TestPatternSync EQUALS RANDOM (0 TO 65535).	
2.1	IF TestPatternSync IS_EQUAL_TO OriginalPatternSync.	
2.1.1	GOTO step 2.	
3	ASSIGN timebaseAscPatternSync EQUALS TestPatternSync.	
4	SET the following objects: timebaseAscPatternSync.	Pass/Fail
5	GET the following objects: timebaseAscPatternSync.	Pass/Fail
6	VERIFY timebaseAscPatternSync IS_EQUAL_TO TestPatternSync.	Pass/Fail
7	ASSIGN timebaseAscPatternSync EQUALS OriginalPatternSync.	
8	SET the following objects: timebaseAscPatternSync.	Pass/Fail
9	GET the following objects: timebaseAscPatternSync.	Pass/Fail
10	VERIFY timebaseAscPatternSync IS_EQUAL_TO OriginalPatternSync.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.10 Configure Pattern to Run Free

Test Procedure:	**Configure Pattern to Run Free**
Description:	**This test case verifies that an ASC allows a management station to configure a pattern for the Free mode when it is called.**
Requirement(s):	• **3.5.2.1.4.1.1 Configure Pattern to Run Free**
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalCycleTime** **patternCycleTime** **TestCycleTime** **patternCycleTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: patternCycleTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalCycleTime.	
4	ASSIGN TestCycleTime EQUALS 0.	
5	ASSIGN patternCycleTime.Table_Row EQUALS TestCycleTime.	
6	SET the following objects: patternCycleTime.Table_Row.	Pass/Fail
7	GET the following objects: patternCycleTime.Table_Row.	Pass/Fail

8	VERIFY patternCycleTime.Table_Row IS_EQUAL_TO TestCycleTime.	Pass/Fail
9	ASSIGN patternCycleTime.Table_Row EQUALS OriginalCycleTime.	
10	SET the following objects: patternCycleTime.Table_Row.	Pass/Fail
11	GET the following objects: patternCycleTime.Table_Row.	Pass/Fail
12	VERIFY patternCycleTime.Table_Row IS_EQUAL_TO OriginalCycleTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.11 Configure Pattern to Run Automatic Flash

Test Procedure:	**Configure Pattern to Run Automatic Flash**	
Description:	**This test case verifies that an ASC allows a management station to configure a pattern for Automatic Flash to be active when it is called.**	
Requirement(s):	• **3.5.2.1.4.1.2 Configure Pattern to Run Automatic Flash**	
Variable(s):	**MaxRows** **Table_Row** **OriginalSequence** **TestSequence**	**maxPatterns** **Int** **patternSequenceNumber** **patternSequenceNumber**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalSequence.	
4	ASSIGN TestSequence EQUALS 0.	
5	ASSIGN patternSequenceNumber.Table_Row EQUALS TestSequence.	
6	SET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
7	GET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
8	VERIFY patternSequenceNumber.Table_Row IS_EQUAL_TO TestSequence.	Pass/Fail
9	ASSIGN patternSequenceNumber.Table_Row EQUALS OriginalSequence.	
10	SET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
11	GET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
12	VERIFY patternSequenceNumber.Table_Row IS_EQUAL_TO OriginalSequence.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.4.12 Configure Pattern Cycle Time for Coordination

Test Procedure:	**Configure Pattern Cycle Time for Coordination**
Description:	**This test case verifies that an ASC allows a management station to configure the length of the cycle for a coordinated timing pattern.**
Requirement(s):	• **3.5.2.1.4.1.3 Configure Pattern Cycle Time for Coordination**
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalCycleTime** **patternCycleTime** **TestCycleTime** **patternCycleTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: patternCycleTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalCycleTime.	
4	ASSIGN TestCycleTime EQUALS RANDOM (1 TO 999).	
4.1	IF TestCycleTime IS_EQUAL_TO OriginalCycleTime.	
4.1.1	GOTO step 4.	
5	ASSIGN patternCycleTime.Table_Row EQUALS TestCycleTime.	
6	SET the following objects: patternCycleTime.Table_Row.	Pass/Fail
7	GET the following objects: patternCycleTime.Table_Row.	Pass/Fail
8	VERIFY patternCycleTime.Table_Row IS_EQUAL_TO TestCycleTime.	Pass/Fail
9	ASSIGN patternCycleTime.Table_Row EQUALS OriginalCycleTime.	
10	SET the following objects: patternCycleTime.Table_Row.	Pass/Fail
11	GET the following objects: patternCycleTime.Table_Row.	Pass/Fail
12	VERIFY patternCycleTime.Table_Row IS_EQUAL_TO OriginalCycleTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.13 Configure Pattern Offset Time

	Configure Pattern Offset Time

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to configure the pattern offset time between the local cycle and the master cycle.**
Requirement(s):	• **3.5.2.1.4.1.4 Configure Pattern Offset Time**
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalOffsetTime** **patternOffsetTime** **TestOffsetTime** **patternOffsetTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: patternOffsetTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOffsetTime.	
4	ASSIGN TestOffsetTime EQUALS RANDOM (0 TO 998).	
4.1	IF TestOffsetTime IS_EQUAL_TO OriginalOffsetTime.	
4.1.1	GOTO step 4.	
5	ASSIGN patternOffsetTime.Table_Row EQUALS TestOffsetTime.	
6	SET the following objects: patternOffsetTime.Table_Row.	Pass/Fail
7	GET the following objects: patternOffsetTime.Table_Row.	Pass/Fail
8	VERIFY patternOffsetTime.Table_Row IS_EQUAL_TO TestOffsetTime.	Pass/Fail
9	ASSIGN patternOffsetTime.Table_Row EQUALS OriginalOffsetTime.	
10	SET the following objects: patternOffsetTime.Table_Row.	Pass/Fail
11	GET the following objects: patternOffsetTime.Table_Row.	Pass/Fail
12	VERIFY patternOffsetTime.Table_Row IS_EQUAL_TO OriginalOffsetTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.14 Configure Pattern Split Association

Test Procedure:	**Configure Pattern Split Association**
Description:	**This test case verifies that the ASC allows a management station to configure the split group to be used by a pattern.**
Requirement(s):	• **3.5.2.1.4.1.5 Configure Pattern Split Association**

Variable(s):	**MaxRows** **Table_Row** **OriginalSplit** **TestSplit** **MaxSplits**	**maxPatterns** **Int** **patternSplitNumber** **patternSplitNumber** **maxSplits**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	GET the following objects: maxSplits.	Pass/Fail
2.1	RECORD this information as MaxSplits.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: patternSplitNumber.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalSplit.	
5	ASSIGN TestSplit EQUALS RANDOM (1 TO MaxSplits).	
5.1	IF TestSplit IS_EQUAL_TO OriginalSplit.	
5.1.1	GOTO step 5.	
6	ASSIGN patternSplitNumber.Table_Row EQUALS TestSplit.	
7	SET the following objects: patternSplitNumber.Table_Row.	Pass/Fail
8	GET the following objects: patternSplitNumber.Table_Row.	Pass/Fail
9	VERIFY patternSplitNumber.Table_Row IS_EQUAL_TO TestSplit.	Pass/Fail
10	ASSIGN patternSplitNumber.Table_Row EQUALS OriginalSplit.	
11	SET the following objects: patternSplitNumber.Table_Row.	Pass/Fail
12	GET the following objects: patternSplitNumber.Table_Row.	Pass/Fail
13	VERIFY patternSplitNumber.Table_Row IS_EQUAL_TO OriginalSplit.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.15 Configure Pattern Sequence Association

Test Procedure:	**Configure Pattern Sequence Association**	
Description:	**This test case verifies that the ASC allows a management station to configure the sequence associated with the pattern.**	
Requirement(s):	• **3.5.2.1.4.1.6 Configure Pattern Sequence Association**	
Variable(s):	**MaxRows** **Table_Row** **OriginalSequence** **TestSequence**	**maxPatterns** **Int** **patternSequenceNumber** **patternSequenceNumber**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPatterns.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalSequence.	
4	ASSIGN TestSequence EQUALS RANDOM (1 TO 255).	
4.1	IF TestSequence IS_EQUAL_TO OriginalSequence.	
4.1.1	GOTO step 5.	
5	ASSIGN patternSequenceNumber.Table_Row EQUALS TestSequence.	
6	SET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
7	GET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
8	VERIFY patternSequenceNumber.Table_Row IS_EQUAL_TO TestSequence.	Pass/Fail
9	ASSIGN patternSequenceNumber.Table_Row EQUALS OriginalSequence.	
10	SET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
11	GET the following objects: patternSequenceNumber.Table_Row.	Pass/Fail
12	VERIFY patternSequenceNumber.Table_Row IS_EQUAL_TO OriginalSequence.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.16 Configure Pattern Maximum Mode

Test Procedure:	**Configure Pattern Maximum Mode**	
Description:	**This test case verifies that the ASC allows a management station to set a maximum mode for a pattern overriding a default setting.**	
Requirement(s):	• **3.5.2.1.4.1.7 Configure Pattern Maximum Mode**	
Variable(s):	**MaxRows** **Table_Row** **OriginalMode** **TestMode**	**maxPatterns** **Int** **patternMaximumMode** **patternMaximumMode**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPatterns.	Pass/Fail

1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: patternMaximumMode.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalMode.	
4	ASSIGN TestMode EQUALS RANDOM (2 TO 6).	
4.1	IF TestMode IS_EQUAL_TO OriginalMode.	
4.1.1	GOTO step 4.	
5	ASSIGN patternMaximumMode.Table_Row EQUALS TestMode.	
6	SET the following objects: patternMaximumMode.Table_Row.	Pass/Fail
7	GET the following objects: patternMaximumMode.Table_Row.	Pass/Fail
8	VERIFY patternMaximumMode.Table_Row IS_EQUAL_TO TestMode.	Pass/Fail
9	ASSIGN patternMaximumMode.Table_Row EQUALS OriginalMode.	
10	SET the following objects: patternMaximumMode.Table_Row.	Pass/Fail
11	GET the following objects: patternMaximumMode.Table_Row.	Pass/Fail
12	VERIFY patternMaximumMode.Table_Row IS_EQUAL_TO OriginalMode.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.17 Configure Pattern Phase Set

Test Procedure:	**Configure Pattern Phase Set**	
Description:	**This test case verifies that the ASC allows a management station to configure the phase set associated with the pattern.**	
Requirement(s):	• **3.5.2.1.4.1.8 Configure Pattern Phase Set**	
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalSet** **patternPhaseSet** **TestSet** **patternPhaseSet** **MaxPhaseSets** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of phase sets the ASC supports'.	
1.1	RECORD this information as MaxPhaseSets.	
2	GET the following objects: maxPatterns.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: patternPhaseSet.Table_Row.	Pass/Fail

4.1	RECORD this information as OriginalSet.	
5	ASSIGN TestSet EQUALS RANDOM (1 TO MaxPhaseSets).	
5.1	IF TestSet IS_EQUAL_TO OriginalSet.	
5.1.1	GOTO step 5.	
6	ASSIGN patternPhaseSet.Table_Row EQUALS TestSet.	
7	SET the following objects: patternPhaseSet.Table_Row.	Pass/Fail
8	GET the following objects: patternPhaseSet.Table_Row.	Pass/Fail
9	VERIFY patternPhaseSet.Table_Row IS_EQUAL_TO TestSet.	Pass/Fail
10	ASSIGN patternPhaseSet.Table_Row EQUALS OriginalSet.	
11	SET the following objects: patternPhaseSet.Table_Row.	Pass/Fail
12	GET the following objects: patternPhaseSet.Table_Row.	Pass/Fail
13	VERIFY patternPhaseSet.Table_Row IS_EQUAL_TO OriginalSet.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.18 Configure Pattern Overlap Set

Test Procedure:	**Configure Pattern Overlap Set**	
Description:	**This test case verifies that the ASC allows a management station to configure the overlap set associated with the pattern.**	
Requirement(s):	• **3.5.2.1.4.1.9 Configure Pattern Overlap Set**	
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalSet** **patternOverlapSet** **TestSet** **patternOverlapSet** **MaxOverlapSets** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'The user shall know the maximum number of overlap sets the ASC supports'.	
1.1	RECORD this information as MaxOverlapSets.	
2	GET the following objects: maxPatterns.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: patternOverlapSet.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalSet.	
5	ASSIGN TestSet EQUALS RANDOM (1 TO MaxOverlapSets).	
5.1	IF TestSet IS_EQUAL_TO OriginalSet.	

5.1.1	GOTO step 5.	
6	ASSIGN patternOverlapSet.Table_Row EQUALS TestSet.	
7	SET the following objects: patternOverlapSet.Table_Row.	Pass/Fail
8	GET the following objects: patternOverlapSet.Table_Row.	Pass/Fail
9	VERIFY patternOverlapSet.Table_Row IS_EQUAL_TO TestSet.	Pass/Fail
10	ASSIGN patternOverlapSet.Table_Row EQUALS OriginalSet.	
11	SET the following objects: patternOverlapSet.Table_Row.	Pass/Fail
12	GET the following objects: patternOverlapSet.Table_Row.	Pass/Fail
13	VERIFY patternOverlapSet.Table_Row IS_EQUAL_TO OriginalSet.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.19 Configure Pattern Vehicle Detector Set

Test Procedure:	**Configure Pattern Vehicle Detector Set**	
Description:	**This test case verifies that the ASC allows a management station to configure the vehicle detector set associated with the pattern.**	
Requirement(s):	• **3.5.2.1.4.1.10 Configure Pattern Vehicle Detector Set**	
Variable(s):	**MaxRows** **Table_Row** **OriginalSet** **TestSet** **MaxVehicleDetectorSets**	**maxPatterns** **Int** **patternVehicleDetectorSet** **patternVehicleDetectorSet** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'The user shall know the maximum number of vehicle detector sets the ASC supports'.	
1.1	RECORD this information as MaxVehicleDetectorSets.	
2	GET the following objects: maxPatterns.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: patternVehicleDetectorSet.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalSet.	
5	ASSIGN TestSet EQUALS RANDOM (1 TO MaxVehicleDetectorSets).	
5.1	IF TestSet IS_EQUAL_TO OriginalSet.	
5.1.1	GOTO step 5.	
6	ASSIGN patternVehicleDetectorSet.Table_Row EQUALS TestSet.	
7	SET the following objects: patternVehicleDetectorSet.Table_Row.	Pass/Fail

8	GET the following objects: patternVehicleDetectorSet.Table_Row.	Pass/Fail
9	VERIFY patternVehicleDetectorSet.Table_Row IS_EQUAL_TO TestSet.	Pass/Fail
10	ASSIGN patternVehicleDetectorSet.Table_Row EQUALS OriginalSet.	
11	SET the following objects: patternVehicleDetectorSet.Table_Row.	Pass/Fail
12	GET the following objects: patternVehicleDetectorSet.Table_Row.	Pass/Fail
13	VERIFY patternVehicleDetectorSet.Table_Row IS_EQUAL_TO OriginalSet.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.20 Configure Pattern Pedestrian Detector Set

Test Procedure:	**Configure Pattern Pedestrian Detector Set**	
Description:	**This test case verifies that the ASC allows a management station to configure the pedestrian detector set associated with the pattern.**	
Requirement(s):	• **3.5.2.1.4.1.11 Configure Pattern Pedestrian Detector Set**	
Variable(s):	**MaxRows** **Table_Row** **OriginalSet** **TestSet** **MaxPedestrianDetectorSets**	**maxPatterns** **Int** **patternPedestrianDetectorSet** **patternPedestrianDetectorSet** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'The user shall know the maximum number of pedestrian detector sets the ASC supports'.	
1.1	RECORD this information as MaxPedestrianDetectorSets.	
2	GET the following objects: maxPatterns.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: patternPedestrianDetectorSet.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalSet.	
5	ASSIGN TestSet EQUALS RANDOM (1 TO MaxPedestrianDetectorSets).	
5.1	IF TestSet IS_EQUAL_TO OriginalSet.	
5.1.1	GOTO step 5.	
6	ASSIGN patternPedestrianDetectorSet.Table_Row EQUALS TestSet.	
7	SET the following objects: patternPedestrianDetectorSet.Table_Row.	Pass/Fail
8	GET the following objects: patternPedestrianDetectorSet.Table_Row.	Pass/Fail

9	VERIFY patternPedestrianDetectorSet.Table_Row IS_EQUAL_TO TestSet.	Pass/Fail
10	ASSIGN patternPedestrianDetectorSet.Table_Row EQUALS OriginalSet.	
11	SET the following objects: patternPedestrianDetectorSet.Table_Row.	Pass/Fail
12	GET the following objects: patternPedestrianDetectorSet.Table_Row.	Pass/Fail
13	VERIFY patternPedestrianDetectorSet.Table_Row IS_EQUAL_TO OriginalSet.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.21 Configure Pattern Special Functions

Test Procedure:	**Configure Pattern Special Functions**
Description:	**This test case verifies that the ASC allows a management station to configure special function outputs to be active when a pattern is active.**
Requirement(s):	• **3.5.2.1.4.1.12 Configure Pattern Special Functions**
Variable(s):	**MaxRows** **maxPatterns** **Table_Row** **Int** **OriginalFunctions** **patternSpecialFunctions** **TestFunctions** **patternSpecialFunctions** **MaxSpecialFunctions** **Int** **MaxValue** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of special functions the ASC supports'.	
1.1	RECORD this information as MaxSpecialFunctions.	
2	ASSIGN MaxValue EQUALS (2 ^ MaxSpecialFunctions – 1)	
3	GET the following objects: maxPatterns.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: patternSpecialFunctions.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalFunctions.	
6	If MaxValue IS_GREATER_THAN 4294967295	
6.1	ASSIGN TestFunctions EQUALS RANDOM (1 TO 4294967295)	
6.1.1	GOTO step 8.	
7	ASSIGN TestFunctions EQUALS RANDOM (1 TO MaxValue)	
8	ASSIGN patternPedestrianDetectorSet.Table_Row EQUALS TestFunctions.	

9	SET the following objects: patternSpecialFunctions.Table_Row.	Pass/Fail
10	GET the following objects: patternSpecialFunctions.Table_Row.	Pass/Fail
11	VERIFY patternSpecialFunctions.Table_Row IS_EQUAL_TO TestFunctions.	Pass/Fail
12	ASSIGN patternSpecialFunctions.Table_Row EQUALS OriginalFunctions.	
13	SET the following objects: patternSpecialFunctions.Table_Row.	Pass/Fail
14	GET the following objects: patternSpecialFunctions.Table_Row.	Pass/Fail
16	VERIFY patternSpecialFunctions.Table_Row IS_EQUAL_TO OriginalFunctions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.22 Determine Maximum Number of Timing Patterns

Test Procedure:	**Determine Maximum Number Timing Patterns**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of patterns that can be configured in the ASC and verifies that the ASC supports the minimum number of patterns required by the user.**	
Requirement(s):	• **3.5.2.1.4.1.13 Determine Maximum Number of Timing Patterns**	
Variable(s):	**UserMinPatterns** **maxPatterns**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of patterns required as specified in FR ID 3.5.2.1.4.1.13 of the PRL.'	
1.1	RECORD this information as UserMinPatterns.	
2	GET the following objects: maxPatterns.	Pass/Fail
3	VERIFY maxPatterns IS_NOT_LESS_THAN UserMinPatterns.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.23 Configure Phase Split Time

Test Procedure:	**Configure Split Time**

Description:	**This test case verifies that the ASC allows a management station to configure the split times.**
Requirement(s):	• **3.5.2.1.5.1.1 Configure Phase Split Time**
Variable(s):	**OriginalSplitTime** **splitTime** **TestSplitTime** **splitTime** **MaxSplits** **maxSplits** **MaxPhases** **maxPhases** **Selected_Phase** **Int** **Selected_Split** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSplits.	Pass/Fail
1.1	RECORD this information as MaxSplits.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxPhases.	
3	ASSIGN Selected_Split EQUALS RANDOM (1 TO MaxSplits).	
4	ASSIGN Selected_Phase EQUALS RANDOM (1 TO MaxPhases).	
5	GET the following objects: splitTime.Selected_Split.Selected_Phase.	Pass/Fail
5.1	RECORD this information as OriginalSplitTime.	
6	ASSIGN TestSplitTime EQUALS RANDOM (0 TO 999).	
6.1	IF TestSplitTime IS_EQUAL_TO OriginalSplitTime.	
6.1.1	GOTO step 6.	
7	ASSIGN Selected_Split.Selected_Phase EQUALS TestSplitTime.	
8	SET the following objects: splitTime.Selected_Split.Selected_Phase.	Pass/Fail
9	GET the following objects: splitTime.Selected_Split.Selected_Phase.	Pass/Fail
10	VERIFY splitTime.Selected_Split.Selected_Phase IS_EQUAL_TO TestSplitTime.	Pass/Fail
11	ASSIGN splitTime.Selected_Split.Selected_Phase EQUALS OriginalSplitTime.	
12	SET the following objects: splitTime.Selected_Split.Selected_Phase.	Pass/Fail
13	GET the following objects: splitTime.Selected_Split.Selected_Phase.	Pass/Fail
14	VERIFY splitTime.Selected_Split.Selected_Phase IS_EQUAL_TO OriginalSplitTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.24 Configure Phase Split Mode

Test Procedure:	**Configure Phase Split Mode**

Description:	**This test case verifies that the ASC allows a management station to configure the recall modes specific to the split group.**
Requirement(s):	• **3.5.2.1.5.1.2 Configure Phase Split Mode**
Variable(s):	**OriginalSplitMode splitMode** **TestSplitMode splitMode** **MaxSplits maxSplits** **MaxPhases maxPhases** **Selected_Split Int** **Selected_Phase Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSplits.	Pass/Fail
1.1	RECORD this information as MaxSplits.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxPhases.	
3	ASSIGN Selected_Split EQUALS RANDOM (1 TO MaxSplits).	
4	ASSIGN Selected_Phase EQUALS RANDOM (1 TO MaxPhases).	
5	GET the following objects: splitMode.Selected_Split.Selected_Phase.	Pass/Fail
5.1	RECORD this information as OriginalSplitMode.	
6	ASSIGN TestSplitTime EQUALS RANDOM (2 TO 8).	
6.1	IF TestSplitTime IS_EQUAL_TO OriginalSplitTime.	
6.1.1	GOTO step 6.	
7	ASSIGN splitMode.Table_Row EQUALS TestSplitMode.	
8	SET the following objects: splitMode.Selected_Split.Selected_Phase.	Pass/Fail
9	GET the following objects: splitMode.Selected_Split.Selected_Phase.	Pass/Fail
10	VERIFY splitMode.Selected_Split.Selected_Phase IS_EQUAL_TO TestSplitMode.	Pass/Fail
11	ASSIGN splitMode.Selected_Split.Selected_Phase EQUALS OriginalSplitMode.	
12	SET the following objects: splitMode.Selected_Split.Selected_Phase.	Pass/Fail
13	GET the following objects: splitMode.Selected_Split.Selected_Phase.	Pass/Fail
14	VERIFY splitMode.Selected_Split.Selected_Phase IS_EQUAL_TO OriginalSplitMode.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.25 Configure Split Coordinated Phase

Test Procedure:	**Configure Split Coordination Phase**

Description:	**This test case verifies that the ASC allows a management station to toggle a phase as a coordinated phase.**
Requirement(s):	• **3.5.2.1.5.1.3 Configure Split Coordination Phase**
Variable(s):	**OriginalSplitCoord** **splitCoordPhase** **TestSplitCoord** **splitCoordPhase** **MaxSplits** **maxSplits** **MaxPhases** **maxPhases** **Selected_Split** **Int** **Selected_Phase** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSplits.	Pass/Fail
1.1	RECORD this information as MaxSplits.	
2	GET the following objects: maxPhases.	Pass/Fail
2.1	RECORD this information as MaxPhases.	
3	ASSIGN Selected_Split EQUALS RANDOM (1 TO MaxSplits).	
4	ASSIGN Selected_Phase EQUALS RANDOM (1 TO MaxPhases).	
5	GET the following objects: splitCoordPhase.Selected_Split.Selected_Phase.	Pass/Fail
5.1	RECORD this information as OriginalSplitCoord.	
6	IF OriginalSplitCoord IS_EQUAL_TO 0.	
6.1	ASSIGN TestSplitCoord EQUALS 1.	
6.2	GOTO step 8.	
7	ASSIGN TestSplitCoord EQUALS 0.	
8	ASSIGN Selected_Split.Selected_Phase EQUALS TestSplitCoord.	
9	SET the following objects: splitCoordPhase.Selected_Split.Selected_Phase.	Pass/Fail
11	GET the following objects: splitCoordPhase.Selected_Split.Selected_Phase.	Pass/Fail
12	VERIFY splitCoordPhase.Selected_Split.Selected_Phase IS_EQUAL_TO TestSplitCoord.	Pass/Fail
13	ASSIGN splitCoordPhase.Selected_Split.Selected_Phase EQUALS OriginalSplitCoord.	
14	SET the following objects: splitCoordPhase.Selected_Split.Selected_Phase.	Pass/Fail
15	GET the following objects: splitCoordPhase.Selected_Split.Selected_Phase.	Pass/Fail
16	VERIFY splitCoordPhase.Selected_Split.Selected_Phase IS_EQUAL_TO OriginalSplitCoord.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.26 Determine Maximum Number of Phase Splits

Test Procedure:	**Determine Maximum Number of Phase Splits**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of phase splits supported by ASC and verifies that the ASC supports the minimum number of phase split groups required by the user.**
Requirement(s):	• **3.5.2.1.5.2 Determine Maximum Number of Phase Splits**
Variable(s):	**UserMinSplits** **maxSplits**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	USER-ACTION 'Determine the number of splits required as specified in FR ID 3.5.2.1.5.2.1 of the PRL.'	
1.1	RECORD this information as UserMinSplits.	
2	GET the following objects: maxSplits.	Pass/Fail
3	VERIFY maxSplits IS_NOT_LESS_THAN UserMinSplits.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.27 Configure Timebased Action - Pattern

Test Procedure:	**Configure Timebased Action - Pattern**
Description:	**This test case verifies that the ASC allows a management station to associate a pattern with an action in the 1202 Time Base Action Table.**
Requirement(s):	• **3.5.2.1.10.1.1 Configure Timebased Action - Pattern**
Variable(s):	**MaxRows** **maxTimebaseAscActions** **Table_Row** **Int** **MaxPatterns** **Int** **OriginalPattern** **timebaseAscPattern** **TestPattern** **timebaseAscPattern**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results

1	PRE-CONDITION 'The user shall know the maximum number of patterns supported by the ASC'.	
1.1	RECORD this information as MaxPatterns.	
2	GET the following objects: maxTimebaseAscActions.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: timebaseAscPattern.	Pass/Fail
4.1	RECORD this information as OriginalPattern.	
5	ASSIGN TestPattern EQUALS RANDOM (1 TO MaxPatterns).	
5.1	IF TestPattern IS_EQUAL_TO OriginalPattern.	
5.1.1	GOTO step 5.	
6	ASSIGN timebaseAscPattern.Table_Row EQUALS TestPattern.	
7	SET the following objects: timebaseAscPattern.Table_Row.	Pass/Fail
8	GET the following objects: timebaseAscPattern.Table_Row.	Pass/Fail
9	VERIFY timebaseAscPattern.Table_Row IS_EQUAL_TO TestPattern.	Pass/Fail
10	ASSIGN timebaseAscPattern.Table_Row EQUALS OriginalPattern.	
11	SET the following objects: timebaseAscPattern.Table_Row.	Pass/Fail
12	GET the following objects: timebaseAscPattern.Table_Row.	Pass/Fail
13	VERIFY timebaseAscPattern.Table_Row IS_EQUAL_TO OriginalPattern.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.28 Configure Timebased Action – Special Functions

Test Procedure:	**Configure Timebased Action – Special Functions**	
Description:	**This test case verifies that the ASC allows a management station to configure special function outputs to be active when a timebased action is active.**	
Requirement(s):	• **3.5.2.1.10.1.2 Configure Timebased Action – Special Functions**	
Variable(s):	**MaxRows** **maxtimebaseAscActions** **Table_Row** **Int** **OriginalFunctions** **timebaseAscSpecialFunction** **TestFunctions** **timebaseAscSpecialFunction** **MaxSpecialFunctions** **Int** **MaxValue** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'The user shall know the maximum number of special functions the ASC supports'.	
1.1	RECORD this information as MaxSpecialFunctions.	

2	ASSIGN MaxValue EQUALS (2 ^ MaxSpecialFunctions – 1)	
3	GET the following objects: maxTimebaseAscActions.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: timebaseAscSpecialFunction.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalFunctions.	
6	If MaxValue IS_GREATER_THAN 4294967295	
6.1	ASSIGN TestFunctions EQUALS RANDOM (1 TO 4294967295)	
6.1.1	GOTO step 8.	
7	ASSIGN TestFunctions EQUALS RANDOM (1 TO MaxValue)	
8	ASSIGN timebaseAscSpecialFunction.Table_Row EQUALS TestFunctions.	
9	SET the following objects: timebaseAscSpecialFunction.Table_Row.	Pass/Fail
10	GET the following objects: timebaseAscSpecialFunction.Table_Row.	Pass/Fail
11	VERIFY timebaseAscSpecialFunction.Table_Row IS_EQUAL_TO TestFunctions.	Pass/Fail
12	ASSIGN timebaseAscSpecialFunction.Table_Row EQUALS OriginalFunctions.	
13	SET the following objects: timebaseAscSpecialFunction.Table_Row.	Pass/Fail
14	GET the following objects: timebaseAscSpecialFunction.Table_Row.	Pass/Fail
16	VERIFY timebaseAscSpecialFunction.Table_Row IS_EQUAL_TO OriginalFunctions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.29 Determine Maximum Number of Timebased Actions

Test Procedure:	**Determine Maximum Number of Timebased Actions**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of time-based action events supported by ASC and verifies that the ASC supports the minimum number of action events required by the user.**
Requirement(s):	• **3.5.2.1.10.1.3 Determine Maximum Number of Timebased Actions**
Variable(s):	**UserMinTimebaseAscActions maxTimebaseAscActions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	USER-ACTION 'Determine the number of action events required as specified in FR ID 3.5.2.1.10.3 of the PRL.'	
1.1	RECORD this information as UserMinTimebaseAscActions.	

2	GET the following objects: maxTimebaseAscActions.	Pass/Fail
3	VERIFY maxTimebaseAscActions IS_NOT_LESS_THAN UserMinTimebaseAscActions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.30 Determine Action In Effect

Test Procedure:	**Determine Action In Effect**
Description:	**This test case verifies that the ASC allows a management station to determine the current time base.**
Requirement(s):	• **3.5.2.1.10.1.4 Determine Action In Effect**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: timebaseAscActionStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.4.31 Activate Action Plan Remotely

Test Procedure:	**Activate Action Plan Remotely**
Description:	**This test case verifies that the ASC allows a management station to activate a configured action plan.**
Requirement(s):	• **3.5.2.1.10.1.4 Determine Action In Effect**
Variable(s):	**OriginalPlanControl actionPlanControl** **TestPlanControl actionPlanControl**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: actionPlanControl.	Pass/Fail
1.1	RECORD this information as OriginalPlanControl.	
2	ASSIGN TestOriginalControl EQUALS RANDOM (0 TO 255).	
2.1	IF TestPlanControl IS_EQUAL_TO OriginalPlanControl	
2.1.1	GOTO step 2	
3	ASSIGN actionPlanControl EQUALS TestPlanControl.	
4	SET the following objects: actionPlanControl.	Pass/Fail
5	GET the following objects: actionPlanControl.	Pass/Fail
6	VERIFY actionPlanControl IS_EQUAL_TO TestPlanControl.	Pass/Fail
7	ASSIGN actionPlanControl EQUALS OriginalPlanControl.	
8	SET the following objects: actionPlanControl.	Pass/Fail
9	GET the following objects: actionPlanControl.	Pass/Fail
10	VERIFY actionPlanControl IS_EQUAL_TO OriginalPlanControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.5 Sequences

C.3.5.1 Configure Sequence Data

Test Procedure:	**Configure Sequence Data**
Description:	**This test case verifies that the ASC allows a management station configure the order of phases within a ring.**
Requirement(s):	• **3.5.2.1.6.1 Configure Sequence Data**
Variable(s):	**MaxSequences** **maxSequences** **Selected_Sequence** **Int** **OriginalData** **sequenceData** **TestData** **sequenceData** **MaxRings** **maxRings** **Selected_Ring** **Int** **MaxPhases** **maxPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum number of phases supported by the ASC'.	
2.1	RECORD this information as MaxPhases.	
3	GET the following objects: maxSequences.	Pass/Fail

3.1	RECORD this information as MaxSequences.	
4	GET the following objects: maxRings.	Pass/Fail
4.1	RECORD this information as MaxRings.	
5	ASSIGN Selected_Sequence EQUALS RANDOM (1 TO MaxSequences).	
6	ASSIGN Selected_Ring EQUALS RANDOM (1 TO MaxRings).	
7	GET the following objects: sequenceData.Selected_Sequence.Selected_Ring.	Pass/Fail
7.1	RECORD this information as OriginalData.	
8	USER-ACTION 'Select a valid value for sequenceData.Selected_Sequence.Selected_Ring. Each octet must not be greater than MaxPhases.'	
9	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
10	ASSIGN sequenceData.Selected_Sequence.Selected_Ring EQUALS TestData.	
11	SET the following objects: sequenceData.Selected_Sequence.Selected_Ring.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
13	GET the following objects: sequenceData.Selected_Sequence.Selected_Ring.	Pass/Fail
14	VERIFY sequenceData.Selected_Sequence.Selected_Ring IS_EQUAL_TO TestData.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
16	ASSIGN sequenceData.Selected_Sequence.Selected_Ring EQUALS OriginalData.	
17	SET the following objects: sequenceData.Selected_Sequence.Selected_Ring.	Pass/Fail
18	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
19	GET the following objects: sequenceData.Selected_Sequence.Selected_Ring.	Pass/Fail
20	VERIFY sequenceData.Selected_Sequence.Selected_Ring IS_EQUAL_TO OriginalData.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.5.2 Determine Maximum Number of Rings

Test Procedure:	**Determine Maximum Number of Rings**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of rings supported.**
Requirement(s):	• **3.5.2.1.6.2 Determine Maximum Number of Rings**

Variable(s):	**UserMinRings maxRings**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of rings required as specified in FR ID 3.5.2.1.6.2 of the PRL.'	
1.1	RECORD this information as UserMinRings.	
2	GET the following objects: maxRings.	Pass/Fail
3	VERIFY maxRings IS_NOT_LESS_THAN UserMinRings.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.5.3 Determine Maximum Number of Sequences

Test Procedure:	**Determine Maximum Number of Sequences**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of sequence plans supported by ASC and verifies that the ASC supports the minimum number of sequence plans required by the user.**	
Requirement(s):	• **3.5.2.1.6.3 Determine Maximum Number of Sequences**	
Variable(s):	**UserMinSequences maxSequences**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of sequence plans required as specified in FR ID 3.5.2.1.6.2.3 of the PRL.'	
1.1	RECORD this information as UserMinSequences.	
2	GET the following objects: maxSequences.	Pass/Fail
3	VERIFY maxSequences IS_NOT_LESS_THAN UserMinSequences.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.6 Channels

C.3.6.1 Configure Channel Control Source

Test Procedure:	**Configure Channel Control Source**
Description:	**This test case verifies that the ASC allows a management station to configure which phase or overlap controls the channel.**
Requirement(s):	• **3.5.2.1.7.1.1 Configure Channel Control Source**
Variable(s):	**MaxChannelRows maxChannels** **Table_Row Int** **OriginalControlSource channelControlSource** **TestControlSource channelControlSource** **MaxControlSource Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannels.	Pass/Fail
1.1	RECORD this information as MaxChannelRows.	
2	GET the following objects: channelControlType.	Pass/Fail
3	IF channelControlType IS_EQUAL_TO 4 OR channelControlType IS_EQUAL_TO 5.	
3.1	GET the following objects: maxOverlaps.	Pass/Fail
3.1.1	RECORD this information as MaxControlSource.	
3.2	GOTO step 5	
4	GET the following objects: maxPhases.	Pass/Fail
4.1	RECORD this information as MaxControlSource.	
5	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxChannelRows).	
6	GET the following objects: channelControlSource.Table_Row.	Pass/Fail
6.1	RECORD this information as OriginalControlSource.	
7	ASSIGN TestControlSource EQUALS RANDOM (0 TO MaxControlSource).	
7.1	IF TestControlSource IS_EQUAL_TO OriginalControlSource.	
7.1.1	GOTO step 7.	
8	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
9	ASSIGN channelControlSource.Table_Row EQUALS TestControlSource.	
10	SET the following objects: channelControlSource.Table_Row.	Pass/Fail
11	GET the following objects: channelControlSource.Table_Row.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
13	VERIFY channelControlSource.Table_Row IS_EQUAL_TO TestControlSource.	Pass/Fail
14	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	

15	ASSIGN channelControlSource.Table_Row EQUALS OriginalControlSource.	
16	SET the following objects: channelControlSource.Table_Row.	Pass/Fail
17	GET the following objects: channelControlSource.Table_Row.	Pass/Fail
18	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
19	VERIFY channelControlSource.Table_Row IS_EQUAL_TO OriginalControlSource.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.6.2 Configure Channel Control Type

Test Procedure:	**Configure Channel Control Type - Vehicle Phase**	
Description:	**This test case verifies that the ASC allows a management station to set channel control type.**	
Requirement(s):	• **3.5.2.1.7.1.2 Configure Channel Control Type**	
Variable(s):	**MaxRows** **Table_Row** **OriginalControlType** **TestControlType**	**maxChannels** **Int** **channelControlType** **channelControlType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded with phase 2 configured with for vehicles and pedestrians'.	
2	GET the following objects: maxChannels.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS 2.	
4	GET the following objects: channelControlType.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalControlType.	
5	ASSIGN TestControlType EQUALS 3.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
6.1	NOTE 3 = phasePedestrian.	
7	ASSIGN channelControlType.Table_Row EQUALS TestControlType.	
8	SET the following objects: channelControlType.Table_Row.	Pass/Fail
9	GET the following objects: channelControlType.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	VERIFY channelControlType.Table_Row IS_EQUAL_TO TestControlType.	Pass/Fail

12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN channelControlType.Table_Row EQUALS OriginalControlType.	
14	SET the following objects: channelControlType.Table_Row.	Pass/Fail
15	GET the following objects: channelControlType.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	VERIFY channelControlType.Table_Row IS_EQUAL_TO OriginalControlType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.6.3 Configure Channel Flash Yellow

Test Procedure:	**Configure Channel Flash Yellow**
Description:	**This test case verifies that the ASC allows a management station to set the channel state for Automatic Flash to Flash Yellow.**
Requirement(s):	• **3.5.2.1.7.1.3.1 Configure Channel Flash Yellow**
Variable(s):	**MaxRows** **maxChannels** **OriginalChannelFlash** **channelFlash** **TestChannelFlash** **channelFlash** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannels.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: channelFlash.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalChannelFlash.	
4	ASSIGN TestChannelFlash EQUALS OriginalChannelFlash AND 251.	
4.1	NOTE 'Toggle Bit 2 LOW because Bits 1 & 2 of channelFlash cannot both be TRUE'.	
5	ASSIGN TestChannelFlash EQUALS TestChannelFlash XOR 2.	
6	ASSIGN channelFlash.Table_Row EQUALS TestChannelFlash.	
7	SET the following objects: channelFlash.Table_Row.	Pass/Fail
8	GET the following objects: channelFlash.Table_Row.	Pass/Fail
9	VERIFY channelFlash.Table_Row IS_EQUAL_TO TestChannelFlash.	Pass/Fail
10	ASSIGN channelFlash.Table_Row EQUALS OriginalChannelFlash.	
11	SET the following objects: channelFlash.Table_Row.	Pass/Fail
12	GET the following objects: channelFlash.Table_Row.	Pass/Fail

13	VERIFY channelFlash.Table_Row IS_EQUAL_TO OriginalChannelFlash.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.6.4 Enable/Disable Channel Flash - Red

Test Procedure:	**Enable/Disable Channel Flash - Red**	
Description:	**This test case verifies that the ASC allows a management station to set the channel state for Automatic Flash to Flash Red.**	
Requirement(s):	• **3.5.2.1.7.1.3.2 Configure Channel Flash Red**	
Variable(s):	**MaxRows** **Table_Row** **OriginalChannelFlash** **TestChannelFlash**	**maxChannels** **Int** **channelFlash** **channelFlash**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxChannels.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: channelFlash.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalChannelFlash.	
4	ASSIGN TestChannelFlash EQUALS OriginalChannelFlash XOR 4.	
5	ASSIGN channelFlash.Table_Row EQUALS TestChannelFlash.	
6	SET the following objects: channelFlash.Table_Row.	Pass/Fail
7	GET the following objects: channelFlash.Table_Row.	Pass/Fail
8	VERIFY channelFlash.Table_Row IS_EQUAL_TO TestChannelFlash.	Pass/Fail
9	ASSIGN channelFlash.Table_Row EQUALS OriginalChannelFlash.	
10	SET the following objects: channelFlash.Table_Row.	Pass/Fail
11	GET the following objects: channelFlash.Table_Row.	Pass/Fail
12	VERIFY channelFlash.Table_Row IS_EQUAL_TO OriginalChannelFlash.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.6.5 Configure Channel Flash Alternate Hertz

	Configure Channel Flash Alternate Hald Hertz

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to set the channel Alternate Half Hertz during Automatic Flash.**
Requirement(s):	• **3.5.2.1.7.1.3.3 Configure Channel Flash Alternate Half Hertz**
Variable(s):	**MaxRows** maxChannels **Table_Row** Int **OriginalChannelFlash** channelFlash **TestChannelFlash** channelFlash
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannels.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: channelFlash.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalChannelFlash.	
4	ASSIGN TestChannelFlash EQUALS OriginalChannelFlash XOR 8.	
5	ASSIGN channelFlash.Table_Row EQUALS TestChannelFlash.	
6	SET the following objects: channelFlash.Table_Row.	Pass/Fail
7	GET the following objects: channelFlash.Table_Row.	Pass/Fail
8	VERIFY channelFlash.Table_Row IS_EQUAL_TO TestChannelFlash.	Pass/Fail
9	ASSIGN channelFlash.Table_Row EQUALS OriginalChannelFlash.	
10	SET the following objects: channelFlash.Table_Row.	Pass/Fail
11	GET the following objects: channelFlash.Table_Row.	Pass/Fail
12	VERIFY channelFlash.Table_Row IS_EQUAL_TO OriginalChannelFlash.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.6.6 Configure Channel Flash Alternate First or Second

Test Procedure:	**Configure Channel Flash Alternate Hald Hertz**
Description:	**This test case verifies that the ASC allows a management station to set the channel to flash first or second in an alternating manner during Automatic Flash.**
Requirement(s):	• **3.5.2.1.7.1.3.4 Configure Channel Flash Alternate Half Hertz**
Variable(s):	**MaxRows** maxChannels **Table_Row** Int

	OriginalChannelFlash channelFlash **TestChannelFlash channelFlash**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxChannels.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: channelFlash.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalChannelFlash.	
4	ASSIGN TestChannelFlash EQUALS OriginalChannelFlash XOR 16.	
5	ASSIGN channelFlash.Table_Row EQUALS TestChannelFlash.	
6	SET the following objects: channelFlash.Table_Row.	Pass/Fail
7	GET the following objects: channelFlash.Table_Row.	Pass/Fail
8	VERIFY channelFlash.Table_Row IS_EQUAL_TO TestChannelFlash.	Pass/Fail
9	ASSIGN channelFlash.Table_Row EQUALS OriginalChannelFlash.	
10	SET the following objects: channelFlash.Table_Row.	Pass/Fail
11	GET the following objects: channelFlash.Table_Row.	Pass/Fail
12	VERIFY channelFlash.Table_Row IS_EQUAL_TO OriginalChannelFlash.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.6.7 Determine Maximum Number of Channels

Test Procedure:	**Determine Maximum Number of Channels Supported**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of channels supported in the channel table.**	
Requirement(s):	• **3.5.2.1.7.2.1 Determine Maximum Number of Channels**	
Variable(s):	**UserMinChannels maxChannels**	
Pass/Fail Criteria:	**The device under test shall pass every verfication step in the test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of channels required as specified in FR ID 3.5.2.1.7.2.1 of the PRL.'	
1.1	RECORD this information as UserMinChannels.	
2	GET the following objects: maxChannels.	Pass/Fail
3	VERIFY maxChannels IS_NOT_LESS_THAN UserMinChannels.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

Test Procedure Notes:

C.3.7 Overlaps

C.3.7.1 Configure Overlap Type – Vehicle Normal

Test Procedure:	**Configure Overlap Type - Normal**	
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to normal.**	
Requirement(s):	• **3.5.2.1.8.1.1.1 Configure Overlap Type - Vehicle Normal**	
Variable(s):	**MaxRows** **Table_Row** **OriginalType** **TestType**	**maxOverlaps** **Int** **overlapType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 2.	
5.1	NOTE 2 = normal.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.2 Configure Overlap Type – Vehicle Minus Green and Yellow

	Configure Overlap Type – Vehicle Minus Green Yellow

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to Minus Green and Yellow**
Requirement(s):	• **3.5.2.1.8.1.1.2 Configure Overlap Type - Vehicle Minus Green and Yellow**
Variable(s):	**MaxRows** **maxOverlaps** **Table_Row** **Int** **OriginalType** **overlapType** **TestType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 3.	
5.1	NOTE 3 = minusGreenYellow.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.3 Configure Overlap Type – Pedestrian Normal

Test Procedure:	**Configure Overlap Type – Pedestrian Normal**

Description:	**This test case verifies that the ASC allows a management station to set the overlap type to pedestrian normal.**
Requirement(s):	• **3.5.2.1.8.1.1.3 Configure Overlap Type – Pedestrian Normal**
Variable(s):	**MaxRows** **maxOverlaps** **Table_Row** **Int** **OriginalType** **overlapType** **TestType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 4.	
5.1	NOTE 4 = pedestrianNormal.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.4 Configure Overlap Type – Flashing Yellow Arrow – 3 Section Head

Test Procedure:	**Configure Overlap Type – Flashing Yellow Arrow – 3 Section Head**
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to a 3 Section Flashing Yellow Arrow.**

Requirement(s):	• **3.5.2.1.8.1.1.4 Configure Overlap Type - Flashing Yellow Arrow - 3 Section Head**
Variable(s):	**MaxRows** **maxOverlaps** **Table_Row** **Int** **OriginalType** **overlapType** **TestType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 5.	
5.1	NOTE 5 = fyaThreeSection.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.5 Configure Overlap Type - Flashing Yellow Arrow - 4 Section Head

Test Procedure:	**Configure Overlap Type - Flashing Yellow Arrow - 4 Section Head**
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to a 4 Section Flashing Yellow Arrow.**
Requirement(s):	• **3.5.2.1.8.1.1.5 Configure Overlap Type - Flashing Yellow Arrow - 4 Section Head**
	MaxRows **maxOverlaps**

Variable(s):	**Table_Row** **OriginalType** **TestType**	**Int** **overlapType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 6.	
5.1	NOTE 6 = fyaFourSection.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.6 Configure Overlap Type - Flashing Red Arrow - 3 Section Head

Test Procedure:	**Configure Overlap Type - Flashing Red Arrow - 3 Section Head**	
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to a 3 Section Flashing Red Arrow.**	
Requirement(s):	• **3.5.2.1.8.1.1.6 Configure Overlap Type - Flashing Red Arrow - 3 Section Head**	
Variable(s):	**MaxRows** **Table_Row** **OriginalType** **TestType**	**maxOverlaps** **Int** **overlapType** **overlapType**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 7.	
5.1	NOTE 7 = fyaThreeSection.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.7 Configure Overlap Type - Flashing Red Arrow - 4 Section Head

Test Procedure:	**Configure Overlap Type - Flashing Red Arrow - 4 Section Head**	
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to a 4 Section Flashing Red Arrow.**	
Requirement(s):	• **3.5.2.1.8.1.1.7 Configure Overlap Type - Flashing Red Arrow - 4 Section Head**	
Variable(s):	**MaxRows** **Table_Row** **OriginalType** **TestType**	**maxOverlaps** **Int** **overlapType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 8.	
5.1	NOTE 8 = fraFourSection.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.8 Configure Overlap Type – 2 Section Transit Specific Signal Head

Test Procedure:	**Configure Overlap Type – 2 Section Transit Specific Signal Head**
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to support a 2-section head.**
Requirement(s):	• **3.5.2.1.8.1.1.8 Configure Overlap Type – 2 Section Transit Specific Signal Head**
Variable(s):	**MaxRows** **maxOverlaps** **Table_Row** **Int** **OriginalType** **overlapType** **TestType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail

1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 9.	
5.1	NOTE 9 = transit-2.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.9 Configure Overlap Type – Minus Green Yellow Alternate

Test Procedure:	**Configure Overlap Type – Minus Green Yellow Alternate**	
Description:	**This test case verifies that the ASC allows a management station to set the overlap type to Vehicle Minus Green and Yellow Alternate.**	
Requirement(s):	• **3.5.2.1.8.1.1.9 Configure Overlap Type – Minus Green Yellow Alternate**	
Variable(s):	**MaxRows** **Table_Row** **OriginalType** **TestType**	**maxOverlaps** **Int** **overlapType** **overlapType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapType.Table_Row.	Pass/Fail

3.1	RECORD this information as OriginalType.	
4	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
5	ASSIGN TestType EQUALS 10.	
5.1	NOTE 10 = minusGreenYellow.	
6	ASSIGN overlapType.Table_Row EQUALS TestType.	
7	SET the following objects: overlapType.Table_Row.	Pass/Fail
8	GET the following objects: overlapType.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	VERIFY overlapType.Table_Row IS_EQUAL_TO TestType.	Pass/Fail
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN overlapType.Table_Row EQUALS OriginalType.	
13	SET the following objects: overlapType.Table_Row.	Pass/Fail
14	GET the following objects: overlapType.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	VERIFY overlapType.Table_Row IS_EQUAL_TO OriginalType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.10 Configure Overlap Included Phases

Test Procedure:	**Configure Overlap Included Phases**	
Description:	**This test case verifies that the ASC allows a management station to configure which phases are included in overlaps.**	
Requirement(s):	• **3.5.2.1.8.1.2 Configure Overlap Included Phases**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhases** **TestPhases**	**maxOverlaps** **Int** **overlapIncludedPhases** **overlapIncludedPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxOverlaps.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: overlapIncludedPhases.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhases.	

5	USER-ACTION 'Select a valid value for overlapIncludedPhases.Table_Row.'	
5.1	RECORD this information as TestPhases.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN overlapIncludedPhases.Table_Row EQUALS TestPhases.	
8	SET the following objects: overlapIncludedPhases.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: overlapIncludedPhases.Table_Row.	Pass/Fail
11	VERIFY overlapIncludedPhases.Table_Row IS_EQUAL_TO TestPhases.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN overlapIncludedPhases.Table_Row EQUALS OriginalPhases.	
14	SET the following objects: overlapIncludedPhases.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: overlapIncludedPhases.Table_Row.	Pass/Fail
17	VERIFY overlapIncludedPhases.Table_Row IS_EQUAL_TO OriginalPhases.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.11 Configure Overlap Modifier Phases

Test Procedure:	**Configure Overlap Modifier Phases**	
Description:	**This test case verifies that the ASC allows a management station to configure a modifier phase for an overlap.**	
Requirement(s):	• **3.5.2.1.8.1.3 Configure Overlap Modifier Phases**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhases** **TestPhases**	**maxOverlaps** **Int** **overlapModifierPhases** **overlapModifierPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	GET the following objects: maxOverlaps.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: overlapModifierPhases.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPhases.	

5	USER-ACTION 'Select a valid value for overlapModifierPhases.Table_Row.'	
5.1	RECORD this information as TestPhases.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN overlapModifierPhases.Table_Row EQUALS TestPhases.	
8	SET the following objects: overlapModifierPhases.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: overlapModifierPhases.Table_Row.	Pass/Fail
11	VERIFY overlapModifierPhases.Table_Row IS_EQUAL_TO TestPhases.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN overlapModifierPhases.Table_Row EQUALS OriginalPhases.	
14	SET the following objects: overlapModifierPhases.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: overlapModifierPhases.Table_Row.	Pass/Fail
17	VERIFY overlapModifierPhases.Table_Row IS_EQUAL_TO OriginalPhases.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.12 Configure Pedestrian Modifier Phases

Test Procedure:	**Configure Pedestrian Modifier Phases**	
Description:	**This test case verifies that the ASC shall allow a management station to configure pedestrian modifier phases for a vehicle overlap.**	
Requirement(s):	• **3.5.2.1.8.1.4 Configure Pedestrian Modifier Phases**	
Variable(s):	**Table_Row** **OriginalConflictingPedPhases** **TestConflictingPedPhases** **MaxRows**	**Int** **overlapConflictingPedPhases** **overlapConflictingPedPhases** **maxOverlaps**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION The ASC supports overlap modifiers.	
2	GET the following objects: maxOverlaps.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: overlapConflictingPedPhases.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalConflictingPedPhases.	

5	USER-ACTION Select a valid value for TestConflictingPedPhases.	
6	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
7	ASSIGN overlapConflictingPedPhases.Table_Row EQUALS TestConflictingPedPhases.	
8	SET the following objects: overlapConflictingPedPhases.Table_Row.	Pass/Fail
9	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
10	GET the following objects: overlapConflictingPedPhases.Table_Row.	Pass/Fail
11	VERIFY overlapConflictingPedPhases.Table_Row IS_EQUAL_TO TestConflictingPedPhases.	Pass/Fail
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN overlapConflictingPedPhases.Table_Row EQUALS OriginalConflictingPedPhases.	
14	SET the following objects: overlapConflictingPedPhases.Table_Row.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
16	GET the following objects: overlapConflictingPedPhases.Table_Row.	Pass/Fail
17	VERIFY overlapConflictingPedPhases.Table_Row IS_EQUAL_TO OriginalConflictingPedPhases.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.13 Configure Overlap Trailing Green

Test Procedure:	**Configure Overlap Trailing Green**	
Description:	**This test case verifies that the ASC allows a management station to configure the trailing green time for an overlap GREEN indication to be extended.**	
Requirement(s):	• **3.5.2.1.8.1.5 Configure Overlap Trailing Green**	
Variable(s):	**MaxRows** maxOverlaps **Table_Row** Int **OriginalTrailGreen** overlapTrailGreen **TestTrailGreen** overlapTrailGreen	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapTrailGreen.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTrailGreen.	
4	ASSIGN TestTrailGreen EQUALS RANDOM (0 TO 255).	

4.1	IF TestTrailGreen IS_EQUAL_TO OriginalTrailGreen.	
4.1.1	GOTO step 4.	
5	ASSIGN overlapTrailGreen.Table_Row EQUALS TestTrailGreen.	
6	SET the following objects: overlapTrailGreen.Table_Row.	Pass/Fail
7	GET the following objects: overlapTrailGreen.Table_Row.	Pass/Fail
8	VERIFY overlapTrailGreen.Table_Row IS_EQUAL_TO TestTrailGreen.	Pass/Fail
9	ASSIGN overlapTrailGreen.Table_Row EQUALS OriginalTrailGreen.	
10	SET the following objects: overlapTrailGreen.Table_Row.	Pass/Fail
11	GET the following objects: overlapTrailGreen.Table_Row.	Pass/Fail
12	VERIFY overlapTrailGreen.Table_Row IS_EQUAL_TO OriginalTrailGreen.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.14 Configure Overlap Trailing Yellow

Test Procedure:	**Configure Overlap Trailing Yellow**	
Description:	**This test case verifies that the ASC allows a management station to configure the trailing yellow time for the overlap in the event that the overlap GREEN indication is extended.**	
Requirement(s):	• **3.5.2.1.8.1.6 Configure Overlap Trailing Yellow**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTrailYellow** **TestTrailYellow**	**maxOverlaps** **Int** **overlapTrailYellow** **overlapTrailYellow**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapTrailYellow.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTrailYellow.	
4	ASSIGN TestTrailYellow EQUALS RANDOM (0 TO 255).	
4.1	IF TestTrailYellow IS_EQUAL_TO OriginalTrailYellow.	
4.1.1	GOTO step 4.	
5	ASSIGN overlapTrailYellow.Table_Row EQUALS TestTrailYellow.	
6	SET the following objects: overlapTrailYellow.Table_Row.	Pass/Fail
7	GET the following objects: overlapTrailYellow.Table_Row.	Pass/Fail
8	VERIFY overlapTrailYellow.Table_Row IS_EQUAL_TO TestTrailYellow.	Pass/Fail
9	ASSIGN overlapTrailYellow.Table_Row EQUALS OriginalTrailYellow.	

10	SET the following objects: overlapTrailYellow.Table_Row.	Pass/Fail
11	GET the following objects: overlapTrailYellow.Table_Row.	Pass/Fail
12	VERIFY overlapTrailYellow.Table_Row IS_EQUAL_TO OriginalTrailYellow.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.15 Configure Overlap Trailing Red Clearance

Test Procedure:	**Configure Overlap Trailing Red Clearance**	
Description:	**This test case verifies that the ASC allows a management station to configure the trailing red clearance time for the overlap in the event that the overlap GREEN indication is extended.**	
Requirement(s):	• **3.5.2.1.8.1.7 Configure Overlap Trailing Red Clearance**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTrailRed** **TestTrailRed**	**maxOverlaps** **Int** **overlapTrailRed** **overlapTrailRed**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: overlapTrailRed.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTrailRed.	
4	ASSIGN TestTrailRed EQUALS RANDOM (0 TO 255).	
4.1	IF TestTrailRed IS_EQUAL_TO OriginalTrailRed.	
4.1.1	GOTO step 4.	
5	ASSIGN overlapTrailRed.Table_Row EQUALS TestTrailRed.	
6	SET the following objects: overlapTrailRed.Table_Row.	Pass/Fail
7	GET the following objects: overlapTrailRed.Table_Row.	Pass/Fail
8	VERIFY overlapTrailRed.Table_Row IS_EQUAL_TO TestTrailRed.	Pass/Fail
9	ASSIGN overlapTrailRed.Table_Row EQUALS OriginalTrailRed.	
10	SET the following objects: overlapTrailRed.Table_Row.	Pass/Fail
11	GET the following objects: overlapTrailRed.Table_Row.	Pass/Fail
12	VERIFY overlapTrailRed.Table_Row IS_EQUAL_TO OriginalTrailRed.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

Test Procedure Notes:

C.3.7.16 Configure Overlap Walk

Test Procedure:	**Configure Overlap Walk**	
Description:	**This test case verifies that the ASC allows a management station to configure the walk time for a pedestrian overlap.**	
Requirement(s):	• **3.5.2.1.8.1.8 Configure Overlap Walk**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOverlapWalk** **TestOverlapWalk**	**maxOverlaps** **Int** **overlapWalk** **overlapWalk**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: overlapWalk.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOverlapWalk.	
4	ASSIGN TestOverlapWalk EQUALS RANDOM (0 TO 255).	
4.1	IF TestOverlapWalk IS_EQUAL_TO OriginalOverlapWalk.	
4.1.1	GOTO step 4.	
5	ASSIGN overlapWalk.Table_Row EQUALS TestOverlapWalk.	
6	SET the following objects: overlapWalk.Table_Row.	Pass/Fail
7	GET the following objects: overlapWalk.Table_Row.	Pass/Fail
8	VERIFY overlapWalk.Table_Row IS_EQUAL_TO TestOverlapWalk.	Pass/Fail
9	ASSIGN overlapWalk.Table_Row EQUALS OriginalOverlapWalk.	
10	SET the following objects: overlapWalk.Table_Row.	Pass/Fail
11	GET the following objects: overlapWalk.Table_Row.	Pass/Fail
12	VERIFY overlapWalk.Table_Row IS_EQUAL_TO OriginalOverlapWalk.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.17 Configure Overlap Pedestrian Clearance

Test Procedure:	**Configure Overlap Pedestrian Clearance**

Description:	**This test case verifies that the ASC allows a management station to configure the pedestrian clearance time for an overlap.**
Requirement(s):	• **3.5.2.1.8.1.9 Configure Overlap Pedestrian Clearance**
Variable(s):	**MaxRows** **maxOverlaps** **Table_Row** **Int** **OriginalOverlapPedClearance** **overlapPedClearance** **TestOverlapPedClearance** **overlapPedClearance**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlaps.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 to MaxRows).	
3	GET the following objects: overlapPedClearance.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOverlapPedClearance.	
4	ASSIGN TestPedAlternateWalk EQUALS RANDOM (0 TO 255).	
4.1	IF TestOverlapPedClearance IS_EQUAL_TO OriginalOverlapPedClearance.	
4.1.1	GOTO step 4.	
5	ASSIGN overlapPedClearance.Table_Row EQUALS TestOverlapPedClearance.	
6	SET the following objects: overlapPedClearance.Table_Row.	Pass/Fail
7	GET the following objects: overlapPedClearance.Table_Row.	Pass/Fail
8	VERIFY overlapPedClearance.Table_Row IS_EQUAL_TO TestOverlapPedClearance.	Pass/Fail
9	ASSIGN overlapPedClearance.Table_Row EQUALS OriginalOverlapPedClearance.	
10	SET the following objects: overlapPedClearance.Table_Row.	Pass/Fail
11	GET the following objects: overlapPedClearance.Table_Row.	Pass/Fail
12	VERIFY overlapPedClearance.Table_Row IS_EQUAL_TO OriginalOverlapPedClearance.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.18 Determine Maximum Number of Overlaps

Test Procedure:	**Determine Maximum Number of Overlaps**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of overlaps supported by ASC and verifies that the ASC supports the minimum number of overlaps required by the user.**

Requirement(s):	• **3.5.2.1.8.3.1 Determine Maximum Number of Overlaps**	
Variable(s):	**UserMinOverlaps** **maxOverlaps**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of overlaps required as specified in FR ID 3.5.2.1.8.3.1 of the PRL.'	
1.1	RECORD this information as UserMinOverlaps.	
2	GET the following objects: maxOverlaps.	Pass/Fail
3	VERIFY maxOverlaps IS_NOT_LESS_THAN UserMinOverlaps.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.7.19 Determine Maximum Number of Overlaps Sets

Test Procedure:	**Determine Maximum Number of Overlaps**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of overlap sets supported by ASC and verifies that the ASC supports the minimum number of overlap sets required by the user.**	
Requirement(s):	• **3.5.2.1.8.3.2 Determine Maximum Number of Overlap Sets**	
Variable(s):	**UserMinOverlapSets** **maxOverlapSets**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of overlaps required as specified in FR ID 3.5.2.1.8.3.2 of the PRL.'	
1.1	RECORD this information as UserMinOverlapSets.	
2	GET the following objects: maxOverlapSets.	Pass/Fail
3	VERIFY maxOverlapSets IS_NOT_LESS_THAN UserMinOverlapSets.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8 Preemption

C.3.8.1 Enable/Disable Preempt Inputs

Test Procedure:	**Enable/Disable Preempt Inputs**
Description:	**This test case verifies that the ASC allows a management station to enable/disable preempt inputs.**
Requirement(s):	• **3.5.2.1.9.1.1 Enable/Disable Preempt Inputs**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalPreemptControl** **preemptControl** **TestPreemptControl** **preemptControl**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControl.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPreemptControl.	
4	ASSIGN TestPreemptControl EQUALS OriginalPreemptControl XOR 16.	
5	ASSIGN preemptControl.Table_Row EQUALS TestPreemptControl.	
6	SET the following objects: preemptControl.Table_Row.	Pass/Fail
7	GET the following objects: preemptControl.Table_Row.	Pass/Fail
8	VERIFY preemptControl.Table_Row IS_EQUAL_TO TestPreemptControl.	Pass/Fail
9	ASSIGN preemptControl.Table_Row EQUALS OriginalPreemptControl.	
10	SET the following objects: preemptControl.Table_Row.	Pass/Fail
11	GET the following objects: preemptControl.Table_Row.	Pass/Fail
12	VERIFY preemptControl.Table_Row IS_EQUAL_TO OriginalPreemptControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.2 Configure Preempt Control - Non-Locking Memory

Test Procedure:	**Configure Preempt Control - Non-Locking Memory**
Description:	**This test case verifies that the ASC allows a management station to configure if a preempt is to not occur if the preempt**

	request terminates prior to the expiration of the preempt delay time.	
Requirement(s):	• **3.5.2.1.9.1.2 Configure Preempt Control - Non-Locking Memory**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPreemptControl** **TestPreemptControl**	**maxPreempts** **Int** **preemptControl** **preemptControl**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControl.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPreemptControl.	
4	ASSIGN TestPreemptControl EQUALS OriginalPreemptControl XOR 1.	
5	ASSIGN preemptControl.Table_Row EQUALS TestPreemptControl.	
6	SET the following objects: preemptControl.Table_Row.	Pass/Fail
7	GET the following objects: preemptControl.Table_Row.	Pass/Fail
8	VERIFY preemptControl.Table_Row IS_EQUAL_TO TestPreemptControl.	Pass/Fail
9	ASSIGN preemptControl.Table_Row EQUALS OriginalPreemptControl.	
10	SET the following objects: preemptControl.Table_Row.	Pass/Fail
11	GET the following objects: preemptControl.Table_Row.	Pass/Fail
12	VERIFY preemptControl.Table_Row IS_EQUAL_TO OriginalPreemptControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.3 Configure Preempt Control – Override Automatic Flash

Test Procedure:	**Configure Preempt Control - Preempt Override Flash**	
Description:	**This test case verifies that the ASC allows a management station to configure if a preempt is not allowed to override automatic flash.**	
Requirement(s):	• **3.5.2.1.9.1.3 Configure Preempt Control - Preempt Override Flash**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPreemptControl** **TestPreemptControl**	**maxPreempts** **Int** **preemptControl** **preemptControl**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControl.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPreemptControl.	
4	ASSIGN TestPreemptControl EQUALS OriginalPreemptControl XOR 2.	
5	ASSIGN preemptControl.Table_Row EQUALS TestPreemptControl.	
6	SET the following objects: preemptControl.Table_Row.	Pass/Fail
7	GET the following objects: preemptControl.Table_Row.	Pass/Fail
8	VERIFY preemptControl.Table_Row IS_EQUAL_TO TestPreemptControl.	Pass/Fail
9	ASSIGN preemptControl.Table_Row EQUALS OriginalPreemptControl.	
10	SET the following objects: preemptControl.Table_Row.	Pass/Fail
11	GET the following objects: preemptControl.Table_Row.	Pass/Fail
12	VERIFY preemptControl.Table_Row IS_EQUAL_TO OriginalPreemptControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.4 Configure Preempt Control – Override Preempt

Test Procedure:	**Configure Preempt Control – Override Preempt**	
Description:	**This test case verifies that the ASC allows a management station to configure if is not allowed to override the preempt next in the preemptTable.**	
Requirement(s):	• **3.5.2.1.9.1.4 Configure Preempt Control – Override Preempt**	
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalPreemptControl** **preemptControl** **TestPreemptControl** **preemptControl**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControl.Table_Row.	Pass/Fail

3.1	RECORD this information as OriginalPreemptControl.	
4	ASSIGN TestPreemptControl EQUALS OriginalPreemptControl XOR 4.	
5	ASSIGN preemptControl.Table_Row EQUALS TestPreemptControl.	
6	SET the following objects: preemptControl.Table_Row.	Pass/Fail
7	GET the following objects: preemptControl.Table_Row.	Pass/Fail
8	VERIFY preemptControl.Table_Row IS_EQUAL_TO TestPreemptControl.	Pass/Fail
9	ASSIGN preemptControl.Table_Row EQUALS OriginalPreemptControl.	
10	SET the following objects: preemptControl.Table_Row.	Pass/Fail
11	GET the following objects: preemptControl.Table_Row.	Pass/Fail
12	VERIFY preemptControl.Table_Row IS_EQUAL_TO OriginalPreemptControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.5 Configure Preempt Control – Flash Dwell

Test Procedure:	**Configure Preempt Control - Flash Dwell**	
Description:	**This test case verifies that the ASC allows a management station for enable or disable the Flash Dwell parameter for a preempt.**	
Requirement(s):	• **3.5.2.1.9.1.5 Configure Preempt Control - Flash Dwell**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPreemptControl** **TestPreemptControl**	**maxPreempts** **Int** **preemptControl** **preemptControl**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControl.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPreemptControl.	
4	ASSIGN TestPreemptControl EQUALS OriginalPreemptControl XOR 8.	
5	ASSIGN preemptControl.Table_Row EQUALS TestPreemptControl.	
6	SET the following objects: preemptControl.Table_Row.	Pass/Fail
7	GET the following objects: preemptControl.Table_Row.	Pass/Fail
8	VERIFY preemptControl.Table_Row IS_EQUAL_TO TestPreemptControl.	Pass/Fail
9	ASSIGN preemptControl.Table_Row EQUALS OriginalPreemptControl.	

10	SET the following objects: preemptControl.Table_Row.	Pass/Fail
11	GET the following objects: preemptControl.Table_Row.	Pass/Fail
12	VERIFY preemptControl.Table_Row IS_EQUAL_TO OriginalPreemptControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.6 Configure Preempt Control – All Red Entry

Test Procedure:	**Configure Preempt Control – All Red Entry**	
Description:	**This test case verifies that the ASC allows a management station for configure the ASC to go to an all-red state and the start of a preempt.**	
Requirement(s):	• **3.5.2.1.9.1.6 Configure Preempt Control – All Red Entry**	
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalPreemptControl** **preemptControl** **TestPreemptControl** **preemptControl**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControl.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPreemptControl.	
4	ASSIGN TestPreemptControl EQUALS OriginalPreemptControl XOR 32.	
5	ASSIGN preemptControl.Table_Row EQUALS TestPreemptControl.	
6	SET the following objects: preemptControl.Table_Row.	Pass/Fail
7	GET the following objects: preemptControl.Table_Row.	Pass/Fail
8	VERIFY preemptControl.Table_Row IS_EQUAL_TO TestPreemptControl.	Pass/Fail
9	ASSIGN preemptControl.Table_Row EQUALS OriginalPreemptControl.	
10	SET the following objects: preemptControl.Table_Row.	Pass/Fail
11	GET the following objects: preemptControl.Table_Row.	Pass/Fail
12	VERIFY preemptControl.Table_Row IS_EQUAL_TO OriginalPreemptControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.7 Configure Preempt Link

Test Procedure:	**Configure Preempt Link**		
Description:	**This test case verifies that the ASC allows a management station to associate preempts with previous preempts in the table.**		
Requirement(s):	• **3.5.2.1.9.1.7 Configure Preempt Link**		
Variable(s):	**MaxRows** **Table_Row** **OriginalPreemptLink** **TestPreemptLink**	**maxPreempts** **Int** **preemptLink** **preemptLink**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (2 TO MaxRows).	
3	GET the following objects: preemptLink.Test_Row.	Pass/Fail
3.1	RECORD this information as OriginalPreemptLink.	
4	ASSIGN TestPreemptLink EQUALS RANDOM (1 TO Table_Row).	
5	ASSIGN preemptLink.Test_Row EQUALS TestPreemptLink.	
6	SET the following objects: preemptLink.Test_Row.	Pass/Fail
7	GET the following objects: preemptLink.Test_Row.	Pass/Fail
8	VERIFY preemptLink.Test_Row IS_EQUAL_TO TestPreemptLink.	Pass/Fail
9	ASSIGN preemptLink.Test_Row EQUALS OriginalPreemptLink.	
10	SET the following objects: preemptLink.Test_Row.	Pass/Fail
11	GET the following objects: preemptLink.Test_Row.	Pass/Fail
12	VERIFY preemptLink.Test_Row IS_EQUAL_TO OriginalPreemptLink.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.8.8 Configure Preempt Delay

Test Procedure:	**Configure Preempt Delay**	
Description:	**This test case verifies that the ASC allows a management station to configure how long a preempt input shall be active prior to any preempt sequence begins.**	
Requirement(s):	• **3.5.2.1.9.1.8 Configure Preempt Delay**	
	MaxRows	**maxPreempts**

Variable(s):	**Table_Row** **OriginalPreemptDelay** **TestPreemptDelay**	**Int** **preemptDelay** **preemptDelay**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: maxPreempts.		Pass/Fail
1.1	RECORD this information as MaxRows.		
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).		
3	GET the following objects: preemptDelay.Table_Row.		Pass/Fail
3.1	RECORD this information as OriginalPreemptDelay.		
4	ASSIGN TestPreemptDelay EQUALS RANDOM (0 TO 600).		
4.1	IF TestPreemptDelay IS_EQUAL_TO OriginalPreemptDelay.		
4.1.1	GOTO step 4.		
5	ASSIGN preemptDelay.Table_Row EQUALS TestPreemptDelay.		
6	SET the following objects: preemptDelay.Table_Row.		Pass/Fail
7	GET the following objects: preemptDelay.Table_Row.		Pass/Fail
8	VERIFY preemptDelay.Table_Row IS_EQUAL_TO TestPreemptDelay.		Pass/Fail
9	ASSIGN preemptDelay.Table_Row EQUALS OriginalPreemptDelay.		
10	SET the following objects: preemptDelay.Table_Row.		Pass/Fail
11	GET the following objects: preemptDelay.Table_Row.		Pass/Fail
12	VERIFY preemptDelay.Table_Row IS_EQUAL_TO OriginalPreemptDelay.		Pass/Fail
Test Procedure Results			
Tested By:		**Date Tested:**	**Pass/Fail**
Test Procedure Notes:			

C.3.8.9 Configure Preempt Minimum Duration

Test Procedure:	**Configure Preempt Minimum Duration**		
Description:	**This test case verifies that the ASC allows a management station to configure the minimum time a preempt shall be active.**		
Requirement(s):	• **3.5.2.1.9.1.9 Configure Preempt Minimum Duration**		
Variable(s):	**MaxRows** **Table_Row** **OriginalMinimumDuration** **TestMinimumDuration**	**maxPreempts** **Int** **preemptMinimumDuration** **preemptMinimumDuration**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**

1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptMinimumDuration.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalMinimumDuration.	
4	ASSIGN TestMinimumDuration EQUALS RANDOM (0 TO 65535).	
4.1	IF TestMinimumDuration IS_EQUAL_TO OriginalMinimumDuration.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptMinimumDuration.Table_Row EQUALS TestMinimumDuration.	
6	SET the following objects: preemptMinimumDuration.Table_Row.	Pass/Fail
7	GET the following objects: preemptMinimumDuration.Table_Row.	Pass/Fail
8	VERIFY preemptMinimumDuration.Table_Row IS_EQUAL_TO TestMinimumDuration.	Pass/Fail
9	ASSIGN preemptMinimumDuration.Table_Row EQUALS OriginalMinimumDuration.	
10	SET the following objects: preemptMinimumDuration.Table_Row.	Pass/Fail
11	GET the following objects: preemptMinimumDuration.Table_Row.	Pass/Fail
12	VERIFY preemptMinimumDuration.Table_Row IS_EQUAL_TO OriginalMinimumDuration.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.10 Configure Preempt Enter Minimum Green Time

Test Procedure:	**Configure Preempt Enter Minimum Green Time**
Description:	**This test case verifies that the ASC allows management station to configure the minimum green time for phases at the start of a preempt.**
Requirement(s):	• **3.5.2.1.9.1.10.1 Configure Preempt Enter Minimum Green Time**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalMinimumGreen** **preemptMinimumGreen** **TestGreenTime** **preemptMinimumGreen**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptMinimumGreen.Table_Row.	Pass/Fail

3.1	RECORD this information as OriginalMinimumGreen.	
4	ASSIGN TestGreenTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestGreenTime IS_NOT_EQUAL_TO OriginalMinimumGreen.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptMinimumGreen.Table_Row EQUALS TestGreenTime.	
6	SET the following objects: preemptMinimumGreen.Table_Row.	Pass/Fail
7	GET the following objects: preemptMinimumGreen.Table_Row.	Pass/Fail
8	VERIFY preemptMinimumGreen.Table_Row IS_EQUAL_TO TestGreenTime.	Pass/Fail
9	ASSIGN preemptMinimumGreen.Table_Row EQUALS OriginalMinimumGreen.	
10	SET the following objects: preemptMinimumGreen.Table_Row.	Pass/Fail
11	GET the following objects: preemptMinimumGreen.Table_Row.	Pass/Fail
12	VERIFY preemptMinimumGreen.Table_Row IS_EQUAL_TO OriginalMinimumGreen.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.11 Configure Preempt Enter Minimum Walk Time

Test Procedure:	**Configure Preempt Enter Minimum Walk Time**	
Description:	**This test case verifies that the ASC allows a management station the minimum walk time for phases at the start of a preempt.**	
Requirement(s):	• **3.5.2.1.9.1.10.2 Configure Preempt Enter Minimum Walk Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalMinimumWalk** **TestMinimumWalk**	**maxPreempts** **Int** **preemptMinimumWalk** **preemptMinimumWalk**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptMinimumWalk.	Pass/Fail
3.1	RECORD this information as OriginalMinimumWalk.	
4	ASSIGN TestMinimumWalk EQUALS RANDOM (0 TO 255).	
4.1	IF TestMinimumWalk IS_EQUAL_TO OriginalMinimumWalk.	
4.1.1	GOTO step 4.	

5	ASSIGN preemptMinimumWalk.Table_Row EQUALS TestMinimumWalk.	
6	SET the following objects: preemptMinimumWalk.Table_Row.	Pass/Fail
7	GET the following objects: preemptMinimumWalk.Table_Row.	Pass/Fail
8	VERIFY preemptMinimumWalk.Table_Row IS_EQUAL_TO TestMinimumWalk.	Pass/Fail
9	ASSIGN preemptMinimumWalk.Table_Row EQUALS OriginalMinimumWalk.	
10	SET the following objects: preemptMinimumWalk.Table_Row.	Pass/Fail
11	GET the following objects: preemptMinimumWalk.Table_Row.	Pass/Fail
12	VERIFY preemptMinimumWalk.Table_Row IS_EQUAL_TO OriginalMinimumWalk.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.12 Configure Preempt Enter Pedestrian Clearance Time

Test Procedure:	**Configure Preempt Enter Pedestrian Clearance Time**
Description:	**This test case verifies that the ASC allows a management station to configure the clearance time for a WALK display that is terminated by a preempt.**
Requirement(s):	• **3.5.2.1.9.1.10.3 Configure Preempt Enter Pedestrian Clearance Time**
Variable(s):	**MaxRows** **Table_Row** **OriginalPedClear** **TestPedClear** **maxPreempts** **Int** **preemptEnterPedClear** **preemptEnterPedClear**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptEnterPedClear.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPedClear.	
4	ASSIGN TestPedClear EQUALS RANDOM (0 TO 255).	
4.1	IF TestPedClear IS_EQUAL_TO OriginalPedClear.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptEnterPedClear.Table_Row EQUALS TestPedClear.	
6	SET the following objects: preemptEnterPedClear.Table_Row.	Pass/Fail
7	GET the following objects: preemptEnterPedClear.Table_Row.	Pass/Fail
8	VERIFY preemptEnterPedClear.Table_Row IS_EQUAL_TO TestPedClear.	Pass/Fail

9	ASSIGN preemptEnterPedClear.Table_Row EQUALS OriginalPedClear.	
10	SET the following objects: preemptEnterPedClear.Table_Row.	Pass/Fail
11	GET the following objects: preemptEnterPedClear.Table_Row.	Pass/Fail
12	VERIFY preemptEnterPedClear.Table_Row IS_EQUAL_TO OriginalPedClear.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.13 Configure Preempt Enter Yellow Change Time

Test Procedure:	**Configure Preempt Enter Yellow Change Time**
Description:	**This test case verifies that the ASC allows a management station to configure the minimum yellow change time for a yellow change interval terminated by a preempt.**
Requirement(s):	• **3.5.2.1.9.1.10.4 Configure Preempt Enter Yellow Change Time**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalYellowChange** **preemptEnterYellowChange** **TestYellowChange** **preemptEnterYellowChange**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptEnterYellowChange.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalYellowChange.	
4	ASSIGN TestYellowChange EQUALS RANDOM (0 TO 255).	
4.1	IF TestYellowChange IS_EQUAL_TO OriginalYellowChange.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptEnterYellowChange.Table_Row EQUALS TestYellowChange.	
6	SET the following objects: preemptEnterYellowChange.Table_Row.	Pass/Fail
7	GET the following objects: preemptEnterYellowChange.Table_Row.	Pass/Fail
8	VERIFY preemptEnterYellowChange.Table_Row IS_EQUAL_TO TestYellowChange.	Pass/Fail
9	ASSIGN preemptEnterYellowChange.Table_Row EQUALS OriginalYellowChange.	
10	SET the following objects: preemptEnterYellowChange.Table_Row.	Pass/Fail
11	GET the following objects: preemptEnterYellowChange.Table_Row.	Pass/Fail
12	VERIFY preemptEnterYellowChange.Table_Row IS_EQUAL_TO OriginalYellowChange.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.14 Configure Preempt Enter Red Clearance Time

Test Procedure:	**Configure Preempt Enter Red Clearance Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the minimum red clearance time for a red clearance interval terminated by a preempt.**	
Requirement(s):	• **3.5.2.1.9.1.10.5 Configure Preempt Enter Red Clearance Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalRedClear** **TestRedClear**	**maxPreempts** **Int** **preemptEnterRedClear** **preemptEnterRedClear**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptEnterRedClear.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalRedClear.	
4	ASSIGN TestRedClear EQUALS RANDOM (0 TO 255).	
4.1	IF TestRedClear IS_EQUAL_TO OriginalRedClear.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptEnterRedClear.Table_Row EQUALS TestRedClear.	
6	SET the following objects: preemptEnterRedClear.Table_Row.	Pass/Fail
7	GET the following objects: preemptEnterRedClear.Table_Row.	Pass/Fail
8	VERIFY preemptEnterRedClear.Table_Row IS_EQUAL_TO TestRedClear.	Pass/Fail
9	ASSIGN preemptEnterRedClear.Table_Row EQUALS OriginalRedClear.	
10	SET the following objects: preemptEnterRedClear.Table_Row.	Pass/Fail
11	GET the following objects: preemptEnterRedClear.Table_Row.	Pass/Fail
12	VERIFY preemptEnterRedClear.Table_Row IS_EQUAL_TO OriginalRedClear.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.15 Configure Preempt Track Clearance Green Time

Test Procedure:	**Configure Preempt Track Clearance Green Time**

Description:	**This test case verifies that the ASC allows a management station to configure the minimum time that preempt track clearance phases display a GREEN indication during the track clearance interval.**
Requirement(s):	• **3.5.2.1.9.1.11.1 Configure Preempt Track Clearance Green Time**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalTrackGreen** **preemptTrackGreen** **TestTrackTime** **preemptTrackGreen**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptTrackGreen.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTrackGreen.	
4	ASSIGN Table_Row EQUALS RANDOM (0 TO 255).	
4.1	IF TestTrackTime IS_EQUAL_TO OriginalTrackGreen.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptTrackGreen.Table_Row EQUALS TestTrackTime.	
6	SET the following objects: preemptTrackGreen.Table_Row.	Pass/Fail
7	GET the following objects: preemptTrackGreen.Table_Row.	Pass/Fail
8	VERIFY preemptTrackGreen.Table_Row IS_EQUAL_TO TestTrackTime.	Pass/Fail
9	ASSIGN preemptTrackGreen.Table_Row EQUALS OriginalTrackGreen.	
10	SET the following objects: preemptTrackGreen.Table_Row.	Pass/Fail
11	GET the following objects: preemptTrackGreen.Table_Row.	Pass/Fail
12	VERIFY preemptTrackGreen.Table_Row IS_EQUAL_TO OriginalTrackGreen.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.16 Configure Preempt Track Clearance Yellow Change Time

Test Procedure:	**Configure Preempt Track Clearance Yellow Change Time**
Description:	**This test case verifies that the ASC allows a management station to configure the minimum time that preempt track clearance phases display track a YELLOW indication during the track clearance interval.**
Requirement(s):	• **3.5.2.1.9.1.11.2 Configure Preempt Track Clearance Yellow Change Time**

Variable(s):	**MaxRows** **maxPreempts** **OriginalYellowChange** **preemptTrackYellowChange** **TestYellowChange** **preemptTrackYellowChange** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptTrackYellowChange.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalYellowChange.	
4	ASSIGN TestYellowChange EQUALS RANDOM (0 TO 255).	
4.1	IF TestYellowChange IS_EQUAL_TO OriginalYellowChange.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptTrackYellowChange.Table_Row EQUALS TestYellowChange.	
6	SET the following objects: preemptTrackYellowChange.Table_Row.	Pass/Fail
7	GET the following objects: preemptTrackYellowChange.Table_Row.	Pass/Fail
8	VERIFY preemptTrackYellowChange.Table_Row IS_EQUAL_TO TestYellowChange.	Pass/Fail
9	ASSIGN preemptTrackYellowChange.Table_Row EQUALS OriginalYellowChange.	
10	SET the following objects: preemptTrackYellowChange.Table_Row.	Pass/Fail
11	GET the following objects: preemptTrackYellowChange.Table_Row.	Pass/Fail
12	VERIFY preemptTrackYellowChange.Table_Row IS_EQUAL_TO OriginalYellowChange.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.17 Configure Preempt Track Red Clearance Time

Test Procedure:	**Configure Preempt Track Red Clearance Time**
Description:	**This test case verifies that the ASC allows a management station to configure the minimum time that preempt track clearance phases display track a RED indication during the track clearance interval.**
Requirement(s):	• **3.5.2.1.9.1.11.3 Configure Preempt Track Red Clearance Time**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalRedClear** **preemptTrackRedClear**

	TestRedClear	**preemptTrackRedClear**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptTrackRedClear.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalRedClear.	
4	ASSIGN TestRedClear EQUALS RANDOM (0 TO 255).	
4.1	IF TestRedClear IS_EQUAL_TO OriginalRedClear.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptTrackRedClear.Table_Row EQUALS TestRedClear.	
6	SET the following objects: preemptTrackRedClear.Table_Row.	Pass/Fail
7	GET the following objects: preemptTrackRedClear.Table_Row.	Pass/Fail
8	VERIFY preemptTrackRedClear.Table_Row IS_EQUAL_TO TestRedClear.	Pass/Fail
9	ASSIGN preemptTrackRedClear.Table_Row EQUALS OriginalRedClear.	
10	SET the following objects: preemptTrackRedClear.Table_Row.	Pass/Fail
11	GET the following objects: preemptTrackRedClear.Table_Row.	Pass/Fail
12	VERIFY preemptTrackRedClear.Table_Row IS_EQUAL_TO OriginalRedClear.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.18 Configure Preempt Track Clearance Phases

Test Procedure:	**Configure Preempt Track Clearance Phases**	
Description:	**This test case verifies that the ASC allows a management station to configure which phases are be active for preempts during the preempt track clearance intervals.**	
Requirement(s):	• **3.5.2.1.9.1.11.4 Configure Preempt Track Clearance Phases**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTrackPhase** **TestTrackPhase** **MaxPhases**	**maxPreempts** **Int** **preemptTrackPhase** **preemptTrackPhase** **maxPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum of phases supported by the ASC'.	
2.1	RECORD this information as MaxPhases.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptTrackPhase.Table_Row.	Pass/Fail
6	USER-ACTION 'Select a valid value for preemptTrackPhase.Table_Row, such as '02 05'. Each phase cannot exceed MaxPhases.'	
6.1	RECORD this information as TestTrackPhase.	
7	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
8	ASSIGN preemptTrackPhase.Table_Row EQUALS TestTrackPhase.	
9	SET the following objects: preemptTrackPhase.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptTrackPhase.Table_Row.	Pass/Fail
12	VERIFY preemptTrackPhase.Table_Row IS_EQUAL_TO TestTrackPhase.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptTrackPhase.Table_Row EQUALS OriginalTrackPhase.	
15	SET the following objects: preemptTrackPhase.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptTrackPhase.Table_Row.	Pass/Fail
18	VERIFY preemptTrackPhase.Table_Row IS_EQUAL_TO OriginalTrackPhase.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.19 Configure Preempt Track Clearance Overlaps

Test Procedure:	**Configure Preempt Track Clearance Overlaps**
Description:	**This test case verifies that the ASC allows a management station configure which overlaps shall be active during a preempt track clearance interval.**
Requirement(s):	• **3.5.2.1.9.1.11.5 Configure Preempt Track Clearance Overlaps**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalTrackOverlap** **preemptTrackOverlap** **TestTrackOverlap** **preemptTrackOverlap**

	MaxOverlaps maxOverlaps	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum of overlaps supported by the ASC'.	
2.1	RECORD this information as MaxOverlaps.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptTrackOverlap.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalTrackOverlap.	
6	USER-ACTION 'Select a valid value for preemptTrackOverlap.Table_Row, such as '01'. Each phase cannot exceed MaxOverlaps.'	
6.1	RECORD this information as TestTrackOverlap.	
7	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
8	ASSIGN preemptTrackOverlap.Table_Row EQUALS TestTrackOverlap.	
9	SET the following objects: preemptTrackOverlap.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptTrackOverlap.Table_Row.	Pass/Fail
12	VERIFY preemptTrackOverlap.Table_Row IS_EQUAL_TO TestTrackOverlap.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptTrackOverlap.Table_Row EQUALS OriginalTrackOverlap.	
15	SET the following objects: preemptTrackOverlap.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptTrackOverlap.Table_Row.	Pass/Fail
18	VERIFY preemptTrackOverlap.Table_Row IS_EQUAL_TO OriginalTrackOverlap.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.20 Configure Preempt Minimum Green Dwell Time

Test Procedure:	**Configure Preempt Minimum Green Dwell Time**
Description:	**This test case verifies that the ASC allows a management station to configure the minimum green time for the preempt dwell interval before phases are allowed to cycle.**

Requirement(s):	• **3.5.2.1.9.1.12.1 Configure Preempt Minimum Green Dwell Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDwellGreen** **TestDwellGreen**	**maxPreempts** **Int** **preemptDwellGreen** **preemptDwellGreen**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptDwellGreen.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDwellGreen.	
4	ASSIGN TestDwellGreen EQUALS RANDOM (0 TO 255).	
4.1	IF TestDwellGreen IS_EQUAL_TO OriginalDwellGreen.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptDwellGreen.Table_Row EQUALS TestDwellGreen.	
6	SET the following objects: preemptDwellGreen.Table_Row.	Pass/Fail
7	GET the following objects: preemptDwellGreen.Table_Row.	Pass/Fail
8	VERIFY preemptDwellGreen.Table_Row IS_EQUAL_TO TestDwellGreen.	Pass/Fail
9	ASSIGN preemptDwellGreen.Table_Row EQUALS OriginalDwellGreen.	
10	SET the following objects: preemptDwellGreen.Table_Row.	Pass/Fail
11	GET the following objects: preemptDwellGreen.Table_Row.	Pass/Fail
12	VERIFY preemptDwellGreen.Table_Row IS_EQUAL_TO OriginalDwellGreen.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.21 Configure Preempt Dwell Phases

Test Procedure:	**Configure Preempt Dwell Phases**	
Description:	**This test case verifies that the ASC allows a management station to configure the phases are to be serviced by a preempt during the preempt dwell interval.**	
Requirement(s):	• **3.5.2.1.9.1.12.2 Configure Preempt Dwell Phases**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDwellPhase** **TestDwellPhase**	**maxPreempts** **Int** **preemptDwellPhase** **preemptDwellPhase**

	MaxPhases maxPhases	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum of phases supported by the ASC'.	
2.1	RECORD this information as MaxPhases.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptDwellPhase.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalDwellPhase.	
6	USER-ACTION 'Select a valid value for preemptDwellPhase.Table_Row such as '02 05'. Each phase cannot exceed MaxPhases.'	
6.1	RECORD this information as TestDwellPhase.	
7	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
8	ASSIGN preemptDwellPhase.Table_Row EQUALS TestDwellPhase.	
9	SET the following objects: preemptDwellPhase.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptDwellPhase.Table_Row.	Pass/Fail
12	VERIFY preemptDwellPhase.Table_Row IS_EQUAL_TO TestDwellPhase.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptDwellPhase.Table_Row EQUALS OriginalDwellPhase.	
15	SET the following objects: preemptDwellPhase.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptDwellPhase.Table_Row.	Pass/Fail
18	VERIFY preemptDwellPhase.Table_Row IS_EQUAL_TO OriginalDwellPhase.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.22 Configure Preempt Pedestrian Movements

Test Procedure:	**Configure Preempt Dwell Pedestrian Movements**
Description:	**This test case verifies that the ASC allows a management station to configure pedestrian movements to dwell in WALK during the preempt dwell interval.**
Requirement(s):	• **3.5.2.1.9.1.12.3 Configure Preempt Pedestrian Movements**

Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalDwellPed** **preemptDwellPed** **TestDwellPed** **preemptDwellPed** **MaxPhases** **maxPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum of phases supported by the ASC'.	
2.1	RECORD this information as MaxPhases.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptDwellPed.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalDwellPed.	
6	USER-ACTION 'Select a valid value for preemptDwellPed.Table_Row, such as '02'. Each phase must not be greater than MaxPhases.'	
7	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
8	ASSIGN preemptDwellPed.Table_Row EQUALS TestDwellPed.	
9	SET the following objects: preemptDwellPed.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptDwellPed.Table_Row.	Pass/Fail
12	VERIFY preemptDwellPed.Table_Row IS_EQUAL_TO TestDwellPed.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptDwellPed.Table_Row EQUALS OriginalDwellPed.	
15	SET the following objects: preemptDwellPed.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptDwellPed.Table_Row.	Pass/Fail
18	VERIFY preemptDwellPed.Table_Row IS_EQUAL_TO OriginalDwellPed.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.23 Configure Preempt Dwell Overlaps

Test Procedure:	**Configure Preempt Dwell Overlaps**
Description:	**This test case verifies that the ASC allows a management station to configure which overlaps shall be active during the preempt dwell interval.**

Requirement(s):	• **3.5.2.1.9.1.12.4 Configure Preempt Dwell Overlaps**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalDwellOverlap** **preemptDwellOverlap** **TestDwellOverlap** **preemptDwellOverlap** **MaxOverlaps** **maxOverlaps**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum number of overlaps supported by the ASC'.	
2.1	RECORD this information as MaxOverlaps.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptDwellOverlap.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalDwellOverlap.	
6	USER-ACTION 'Select a valid value for preemptDwellOverlap.Table_Row, such as '01'. Each overlap must not be greater than MaxOverlaps.'	
6.1	RECORD this information as TestDwellOverlap.	
7	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
8	ASSIGN preemptDwellOverlap.Table_Row EQUALS TestDwellOverlap.	
9	SET the following objects: preemptDwellOverlap.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptDwellOverlap.Table_Row.	Pass/Fail
12	VERIFY preemptDwellOverlap.Table_Row IS_EQUAL_TO TestDwellOverlap.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptDwellOverlap.Table_Row EQUALS OriginalDwellOverlap.	
15	SET the following objects: preemptDwellOverlap.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptDwellOverlap.Table_Row.	Pass/Fail
18	VERIFY preemptDwellOverlap.Table_Row IS_EQUAL_TO OriginalDwellOverlap.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.24 Configure Preempt Cycling Phases

Test Procedure:	**Configure Preempt Cycling Phases**
Description:	**This test case verifies that the ASC allows a management station to configure which phases are allowed to cycle during the preempt dwell interval.**
Requirement(s):	• **3.5.2.1.9.1.12.5 Configure Preempt Cycling Phases**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalCyclingPhase** **preemptCyclingPhase** **TestCyclingPhase** **preemptCyclingPhase** **MaxPhases** **maxPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know maximum number of phases supported by the ASC'.	
2.1	RECORD this information as MaxPhases.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptCyclingPhase.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalCyclingPhase.	
6	USER-ACTION 'Select a valid value for preemptCyclingPhase.Table_Row, such as '02'. Each phase must not be greater than MaxPhases.'	
6.1	RECORD this information as TestCyclingPhase.	
7	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
8	ASSIGN preemptCyclingPhase.Table_Row EQUALS TestCyclingPhase.	
9	SET the following objects: preemptCyclingPhase.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptCyclingPhase.Table_Row.	Pass/Fail
12	VERIFY preemptCyclingPhase.Table_Row IS_EQUAL_TO TestCyclingPhase.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptCyclingPhase.Table_Row EQUALS OriginalCyclingPhase.	
15	SET the following objects: preemptCyclingPhase.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptCyclingPhase.Table_Row.	Pass/Fail

18	VERIFY preemptCyclingPhase.Table_Row IS_EQUAL_TO OriginalCyclingPhase.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.25 Configure Preempt Cycling Pedestrian Movements

Test Procedure:	**Configure Preempt Cycling Pedestrian Movements**	
Description:	**This test case verifies that the ASC allows a management station to configure which phases with pedestrian movements are allowed to cycle during the preempt dwell interval.**	
Requirement(s):	• **3.5.2.1.9.1.12.6 Configure Preempt Dwell Overlaps**	
Variable(s):	**MaxRows** **Table_Row** **OriginalCyclingPed** **TestCyclingPed** **MaxPhases**	**maxPreempts** **Int** **preemptCyclingPed** **preemptCyclingPed** **maxPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum number of phases supported by the ASC'.	
2.1	RECORD this information as MaxPhases.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptCyclingPed.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalCyclingPed.	
6	USER-ACTION 'Select a valid value for preemptCyclingPed.Table_Row, such as '02'. Each phase must not be greater than MaxPhases.'	
6.1	RECORD this information as TestCyclingPed.	
7	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
8	ASSIGN preemptCyclingPed.Table_Row EQUALS TestCyclingPed.	
9	SET the following objects: preemptCyclingPed.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptCyclingPed.Table_Row.	Pass/Fail
12	VERIFY preemptCyclingPed.Table_Row IS_EQUAL_TO TestCyclingPed.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptCyclingPed.Table_Row EQUALS OriginalCyclingPed.	

15	SET the following objects: preemptCyclingPed.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error".	
17	GET the following objects: preemptCyclingPed.Table_Row.	Pass/Fail
18	VERIFY preemptCyclingPed.Table_Row IS_EQUAL_TO OriginalCyclingPed.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.26 Configure Preempt Cycling Phases Sequence

Test Procedure:	**Configure Preempt Cycling Phases Sequence**
Description:	**This test case verifies that the ASC allows a management station to configure the sequence for phases to cycle during a preempt dwell interval after the minimum green dwell time.**
Requirement(s):	• **3.5.2.1.9.1.12.7 Configure Preempt Cycling Phases Sequence**
Variable(s):	**MaxPreempts** **maxPreempts** **MaxSequences** **maxSequences** **Table_Row** **Int** **OriginalSequence** **preemptSequenceNumber** **TestSequence** **preemptSequenceNumber**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxPreempts.	
2	GET the following objects: maxSequences.	Pass/Fail
2.1	RECORD this information as MaxSequences.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxPreempts).	
3	GET the following objects: preemptSequenceNumber.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalSequence.	
4	ASSIGN TestSequence EQUALS RANDOM (0 TO MaxSequences).	
4.1	IF TestSequence IS_EQUAL_TO OrignalSequence.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptSequenceNumber.Table_Row EQUALS TestSequence.	
6	SET the following objects: preemptSequenceNumber.Table_Row.	Pass/Fail
7	GET the following objects: preemptSequenceNumber.Table_Row.	Pass/Fail
8	VERIFY preemptSequenceNumber.Table_Row IS_EQUAL_TO TestSequence.	Pass/Fail
9	ASSIGN preemptSequenceNumber.Table_Row EQUALS OriginalSequence.	

10	SET the following objects: preemptSequenceNumber.Table_Row.	Pass/Fail
11	GET the following objects: preemptSequenceNumber.Table_Row.	Pass/Fail
12	VERIFY preemptSequenceNumber.Table_Row IS_EQUAL_TO Original Sequence.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.27 Configure Preempt Cycling Overlaps

Test Procedure:	**Configure Preempt Cycling Overlaps**
Description:	**This test case verifies that the ASC allows a management station to configure which overlaps are allowed to cycle during the preempt dwell interval.**
Requirement(s):	• **3.5.2.1.9.1.12.8 Configure Preempt Cycling Overlaps**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int** **OriginalCyclingOverlap** **preemptCyclingOverlap** **TestCyclingOverlap** **preemptCyclingOverlap** **MaxOverlaps** **maxOverlaps**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum number of overlaps supported by the ASC'.	
2.1	RECORD this information as MaxOverlaps.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptCyclingOverlap.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalCyclingOverlap.	
6	USER-ACTION 'Select a valid value for preemptDwellOverlap.Table_Row, such as '01'. Each overlap must not be greater than MaxOverlaps.'	
6.1	RECORD this information as TestCyclingOverlap.	
7	PERFORM the Test Procedure "C.3.1.1 Create Database Transaction".	
8	ASSIGN preemptCyclingOverlap.Table_Row EQUALS TestCyclingOverlap.	
9	SET the following objects: preemptCyclingOverlap.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptCyclingOverlap.Table_Row.	Pass/Fail

12	VERIFY preemptCyclingOverlap.Table_Row IS_EQUAL_TO TestCyclingOverlap.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptCyclingOverlap.Table_Row EQUALS OriginalCyclingOverlap.	
15	SET the following objects: preemptCyclingOverlap.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptCyclingOverlap.Table_Row.	Pass/Fail
18	VERIFY preemptCyclingOverlap.Table_Row IS_EQUAL_TO OriginalCyclingOverlap.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.28 Configure Preempt Exit Phases

Test Procedure:	**Configure Preempt Exit Phases**	
Description:	**This test cases verifies that the ASC allows a management station to configure the phases to be serviced after a preempt dwell interval.**	
Requirement(s):	• **3.5.2.1.9.1.13.1 Configure Preempt Exit Phases**	
Variable(s):	**MaxRows** **Table_Row** **OriginalExitPhase** **TestExitPhase** **MaxPhases**	**maxPreempts** **Int** **preemptExitPhase** **preemptExitPhase** **maxPhases**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'A valid timing plan is loaded'.	
2	PRE-CONDITION 'The user shall know the maximum number of phases supported by the ASC'.	
2.1	RECORD this information as MaxPhases.	
3	GET the following objects: maxPreempts.	Pass/Fail
3.1	RECORD this information as MaxRows.	
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: preemptExitPhase.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalExitPhase.	
6	USER-ACTION 'Select a valid value for preemptExitPhase.Table_Row, such as '02 06'. Each phase cannot exceed MaxPhases.'	
6.1	RECORD this information as TestExitPhase.	
7	PERFORM the Test Procedure "C.3.1.1 Create Database Transaction".	

8	ASSIGN preemptExitPhase.Table_Row EQUALS TestExitPhase.	
9	SET the following objects: preemptExitPhase.Table_Row.	Pass/Fail
10	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
11	GET the following objects: preemptExitPhase.Table_Row.	Pass/Fail
12	VERIFY preemptExitPhase.Table_Row IS_EQUAL_TO TestExitPhase.	Pass/Fail
13	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
14	ASSIGN preemptExitPhase.Table_Row EQUALS OriginalExitPhase.	
15	SET the following objects: preemptExitPhase.Table_Row.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: preemptExitPhase.Table_Row.	Pass/Fail
18	VERIFY preemptExitPhase.Table_Row IS_EQUAL_TO OriginalExitPhase.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.29 Configure Preempt Exit Phase Strategy

Test Procedure:	**Configure Preempt Exit Phase Strategy**	
Description:	**This test case verifies that the ASC allows a management station to configure the ASC to resume a coordination cycle upon exiting a preempt.**	
Requirement(s):	• **3.5.2.1.9.1.13.2 Configure Preempt Exit Phase Strategy**	
Variable(s):	**MaxRows** **Table_Row** **OriginalExitType** **TestExitType**	**maxPreempts** **Int** **preemptExitType** **preemptExitType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptExitType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalExitType.	
4	ASSIGN TestExitType EQUALS RANDOM (1 TO 4).	
4.1	IF TestExitType IS_EQUAL_TO OriginalExitType.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptExitType.Table_Row _Row EQUALS TestExitType.	
6	SET the following objects: preemptExitType.Table_Row.	Pass/Fail

7	GET the following objects: preemptExitType.Table_Row.	Pass/Fail
8	VERIFY preemptExitType.Table_Row IS_EQUAL_TO TestExitType.	Pass/Fail
9	ASSIGN preemptExitType.Table_Row _Row EQUALS OriginalExitType.	
10	SET the following objects: preemptExitType.Table_Row.	Pass/Fail
11	GET the following objects: preemptExitType.Table_Row.	Pass/Fail
12	VERIFY preemptExitType.Table_Row IS_EQUAL_TO OriginalExitType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.30 Configure Preempt Exit Priority Levels

Test Procedure:	**Configure Preempt Exit Priority Levels**	
Description:	**This test case verifies that the ASC allows a management station to associate a priority level with a detector for a Queue Delay Recovery exit strategy during a preempt.**	
Requirement(s):	• **3.5.2.1.9.1.13.3 Configure Preempt Exit Priority Levels**	
Variable(s):	**MaxPreempts** **maxPreempts** **OriginalDetectorWeight** **preemptDetectorWeight** **TestDetectorWeight** **preemptDetectorWeight** **MaxVehicleDetectors** **maxVehicleDetectors** **Selected_Preempt** **Int** **Selected_Detector** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxPreempts.	
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxVehicleDetectors.	
2	ASSIGN Selected_Preempt EQUALS RANDOM (1 TO MaxPreempts).	
2	ASSIGN Selected_Detector EQUALS RANDOM (1 TO MaxVehicleDetectors).	
3	GET the following objects: preemptDetectorWeight.SelectedPreempt.Selected.Detector.	Pass/Fail
3.1	RECORD this information as OriginalDetectorWeight.	
4	ASSIGN TestDetectorWeight EQUALS RANDOM (0 TO 1000).	
4.1	IF TestDetectorWeight IS_EQUAL_TO OriginalDetectorWeight.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptDetectorWeight.SelectedPreempt.Selected.Detector EQUALS TestDetectorWeight.	

6	SET the following objects: preemptDetectorWeight.SelectedPreempt.Selected.Detector.	Pass/Fail
7	GET the following objects preemptDetectorWeight.SelectedPreempt.Selected.Detector.	Pass/Fail
8	VERIFY preemptDetectorWeight.SelectedPreempt.Selected.Detector IS_EQUAL_TO TestDetectorWeight.	Pass/Fail
9	ASSIGN preemptDetectorWeight.SelectedPreempt.Selected.Detector EQUALS OriginalDetectorWeight.	
10	SET the following objects: preemptDetectorWeight.SelectedPreempt.Selected.Detector.	Pass/Fail
11	GET the following objects: preemptDetectorWeight.SelectedPreempt.Selected.Detector.	Pass/Fail
12	VERIFY preemptDetectorWeight.SelectedPreempt.Selected.Detector IS_EQUAL_TO OriginalDetectorWeight.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.31 Configure Preempt Maximum Presence Time

Test Procedure:	**Configure Preempt Maximum Presence Time**		
Description:	**This test case verifies that the ASC allows a management station to configure the maximum time a preempt call may remain active and be considered valid.**		
Requirement(s):	• **3.5.2.1.9.1.14.1 Configure Preempt Maximum Presence Time**		
Variable(s):	**MaxRows** **Table_Row** **OriginalMaxPresence** **TestMaxPresence**	**maxPreempts** **Int** **preemptMaximumPresence** **preemptMaximumPresence**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptMaximumPresence.	Pass/Fail
3.1	RECORD this information as OriginalMaxPresence.	
4	ASSIGN TestMaxPresence EQUALS RANDOM (0 TO 65535).	
4.1	IF TestMaxPresence IS_EQUAL_TO OriginalMaxPresence.	
4.1.1	GOTO step 4.	
5	ASSIGN preemptMaximumPresence.Table_Row EQUALS TestMaxPresence.	
6	SET the following objects: preemptMaximumPresence.Table_Row.	Pass/Fail

7	GET the following objects: preemptMaximumPresence.Table_Row.	Pass/Fail
8	VERIFY preemptMaximumPresence.Table_Row IS_EQUAL_TO TestMaxPresence.	Pass/Fail
9	ASSIGN preemptMaximumPresence.Table_Row EQUALS OriginalMaxPresence.	
10	SET the following objects: preemptMaximumPresence.Table_Row.	Pass/Fail
11	GET the following objects: preemptMaximumPresence.Table_Row.	Pass/Fail
12	VERIFY preemptMaximumPresence.Table_Row IS_EQUAL_TO OriginalMaxPresence.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.32 Configure Preempt Maximum Presence Action

Test Procedure:	**Configure Preempt Maximum Prescence Action**	
Description:	**This test case verifies that the ASC allows a management station to configure if an all-red flash action is executed after the preempt's maximum presence time or if it services exit phases.**	
Requirement(s):	• **3.5.2.1.9.1.14.2 Configure Preempt Max Presence Action**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPreemptControl** **TestPreemptControl**	**maxPreempts** **Int** **preemptControl** **preemptControl**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControl.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPreemptControl.	
4	ASSIGN TestPreemptControl EQUALS OriginalPreemptControl XOR 32.	
5	ASSIGN preemptControl.Table_Row EQUALS TestPreemptControl.	
6	SET the following objects: preemptControl.Table_Row.	Pass/Fail
7	GET the following objects: preemptControl.Table_Row.	Pass/Fail
8	VERIFY preemptControl.Table_Row IS_EQUAL_TO TestPreemptControl.	Pass/Fail
9	ASSIGN preemptControl.Table_Row EQUALS OriginalPreemptControl.	
10	SET the following objects: preemptControl.Table_Row.	Pass/Fail
11	GET the following objects: preemptControl.Table_Row.	Pass/Fail

12	VERIFY preemptControl.Table_Row IS_EQUAL_TO OriginalPreemptControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.33 Configure Preempt Gate Description

Test Procedure:	**Configure Preempt Gate Description**	
Description:	**This test case verifies that the ASC allows a management station to configure descriptions for preempt gates.**	
Requirement(s):	• **3.5.2.1.9.1.15 Configure Preempt Gate Description**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDescription**	**maxPreempts** **Int** **preemptGateDescription**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptGateDescription.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDescription	
4	ASSIGN preemptGateDescription.Table_Row EQUALS 'Test'.	
5	SET the following objects: preemptGateDescription.Table_Row.	Pass/Fail
6	GET the following objects: preemptGateDescription.Table_Row.	Pass/Fail
7	VERIFY preemptGateDescription.Table_Row IS_EQUAL_TO 'Test'.	Pass/Fail
8	ASSIGN preemptGateDescription.Table_Row EQUALS OrignalDescription.	
9	SET the following objects: preemptGateDescription.Table_Row.	Pass/Fail
10	GET the following objects: preemptGateDescription.Table_Row.	Pass/Fail
11	VERIFY preemptGateDescription.Table_Row IS_EQUAL_TO OriginalDescription.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.8.34 Determine Maximum Number of Preempts

Test Procedure:	**Determine Maximum Number of Preempts**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of preempts**

	supported by ASC and verifies that the ASC supports the minimum number of preempts required by the user.	
Requirement(s):	• **3.5.2.1.9.2 Determine Maximum Number of Preempts**	
Variable(s):	**UserMinPreempts** **maxPreempts**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of preempts required as specified in FR ID 3.5.2.1.9.2 of the PRL.'	
1.1	RECORD this information as UserMinPreempts.	
2	GET the following objects: maxPreempts.	Pass/Fail
3	VERIFY maxPreempts IS_NOT_LESS_THAN UserMinPreempts.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9 Communication and IO

C.3.9.1 Determine Number of ASC Communications Ports

Test Procedure:	**Determine Number of ASC Communications Ports**	
Description:	**This test case verifies that the ASC allows a management station to determine the number of physical communications ports on the device.**	
Requirement(s):	• **3.5.1.2.2.1 Determine Number of ASC Communications Ports**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxCommPorts	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.2 Retrieve Maximum Number of I/O Maps

Test Procedure:	**Retrieve Maximum Number of I/O Maps**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of I/O maps supported.**	
Requirement(s):	• **3.5.2.1.11.2.1 Retrieve Maximum Number I/O Maps**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: asclOmaxMaps.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.3 Retrieve Maximum Number of I/O Map Inputs

Test Procedure:	**Retrieve Maximum Number of I/O Map Inputs**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of I/O map inputs supported.**	
Requirement(s):	• **3.5.2.1.11.2.2 Retrieve Maximum Number I/O Maps Inputs**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: asclOmapMaxInputs.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.4 Retrieve Maximum Number of I/O Map Outputs

Test Procedure:	**Retrieve Maximum Number of I/O Map Outputs**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of I/O map outputs supported.**
Requirement(s):	• **3.5.2.1.11.2.3 Retrieve Maximum Number I/O Maps Outputs**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: asclOmapMaxOutputs.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.5 Retrieve I/O Mapping Activate Conditions

Test Procedure:	**Retrieve I/O Mapping Activate Conditions**
Description:	**This test case verifies that the ASC allows a management station to determine the conditions for when a new I/O map takes effect.**
Requirement(s):	• **3.5.2.1.11.2.4 Retrieve I/O Mapping Activate Conditions**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: asclOactivateRequirement	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.6 Retrieve I/O Mapping Input Functions

Test Procedure:	**Retrieve I/O Mapping Input Functions**	
Description:	**This test case verifies that the ASC allows a management station to view the I/O Mapping input Functions.**	
Requirement(s):	• **3.5.2.1.11.2.5 Retrieve I/O Mapping Input Functions**	
Variable(s):	**MaxRows** **Table_Row**	**asclOmapMaxInputFunctions** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: asclOmapMaxInputFunctions.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: asclOinputMaxFuncIndex.Table_Row, asclOinputFunctionName.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.7 Retrieve I/O Mapping Output Functions

Test Procedure:	**Retrieve I/O Mapping Output Functions**	
Description:	**This test case verifies that the ASC allows a management station to view the I/O Mapping output Functions.**	
Requirement(s):	• **3.5.2.1.11.2.6 Retrieve I/O Mapping Output Functions**	
Variable(s):	**MaxRows** **Table_Row**	**asclOmapMaxOutputFunctions** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: asclOmapMaxOutputFunctions.	Pass/Fail
1.1	RECORD this information as MaxRows.	

2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: asclOoutputMaxFuncIndex.Table_Row, asclOoutputFunctionName.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.8 Retrieve I/O Map Input Device Pin Status

Test Procedure:	**Retrieve I/O Map Input Device Pin Status**	
Description:	**This test case verifies that the ASC allows a management station to view the status of I/O map input device pin.**	
Requirement(s):	• **3.5.2.1.11.2.7 Retrieve I/O Map Input Device Pin Status**	
Variable(s):	**MaxMaps** **asclOmaxMaps** **MaxInputs** **asclOmapMaxInputs** **Map** **Int** **Input** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: asclOmaxMaps.	Pass/Fail
1.1	RECORD this information as MaxMaps.	
2	GET the following objects: asclOmapMaxInputs.	Pass/Fail
2.1	RECORD this information as MaxInputs.	
2	FOR Map from 1 to MaxMaps.	
2.1	FOR Input from 1 to MaxInputs.	
2.1.1	GET the following objects: asclOinputMapDevPinDesc.Map.Input, asclOinputMapDevPinStatus.Map.Input.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.9 Retrieve I/O Map Output Device Pin Status

Test Procedure:	**Retrieve I/O Map Output Device Pin Status**
Description:	**This test case verifies that the ASC allows a management station to view the status of I/O map output device pin.**

Requirement(s):	• **3.5.2.1.11.2.8 Retrieve I/O Map Output Device Pin Status**
Variable(s):	**MaxMaps** **asclOmaxMaps** **MaxOutputs** **asclOmapMaxOutputs** **Map** **Int** **Output** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: asclOmaxMaps.	Pass/Fail
1.1	RECORD this information as MaxMaps.	
2	GET the following objects: asclOmapMaxOutputs.	Pass/Fail
2.1	RECORD this information as MaxOutputs.	
2	FOR Map from 1 to MaxMaps.	
2.1	FOR Output from 1 to MaxOutputs.	
2.1.1	GET the following objects: asclOoutputMapDevPinDesc.Map.Output, asclOoutputMapDevPinStatus.Map.Output.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.10 Determine Serial Bus 1 Device Present

Test Procedure:	**Determine Serial Bus 1 Device Present**
Description:	**This test case verifies that the ASC allows a management station to identify if a device is present for a Serial Bus 1 address in an ATC cabinet.**
Requirement(s):	• **3.5.2.1.12.1.1 Determine Serial Bus 1 Device Present**
Variable(s):	**MaxRows** **maxSIUPort1Addresses** **Table_Row** **Int** **UserMinSerialBus1Addresses** **maxSIUPort1Addresses**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	USER-ACTION Determine the number of channels required as specified in FR ID 3.5.2.1.12.1.1 of the PRL.	
1.1	RECORD this information as UserMinSerialBus1Addresses	
2	GET the following objects: maxSIUPort1Addresses.	Pass/Fail

2.1	RECORD this information as MaxRows.	
3	VERIFY MaxRows IS_NOT_LESS_THAN UserMinSerialBus1Addresses	Pass/Fail
4	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
5	GET the following objects: siuport1Number.Table_Row, siuPort1DevicePresent.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.11 Determine TS2 Port 1 Device Present

Test Procedure:	**Determine TS2 Port 1 Device Present**		
Description:	**This test case verifies that the ASC allows a management station to identify if a device is present for a TS2 Port 1.**		
Requirement(s):	• **3.5.2.1.12.2.1 Determine TS2 Port 1 Device Present**		
Variable(s):	**MaxRows** **Table_Row**	**maxPort1Addresses** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPort1Addresses.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: port1Number.Table_Row, port1DevicePresent.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.12 Enable/Disable TS2 Port 1 Frame 40 Messages

Test Procedure:	**Enable/Disable TS2 Port 1 Frame 40 Messages**
Description:	**This test case verifies that the ASC allows a management station to enable or disable TS2 Port 1 Frame 40 messages.**
Requirement(s):	• **3.5.2.1.12.2.2 Enable/Disable TS2 Port 1 Frame 40 Messages**

Variable(s):	**MaxRows** **Table_Row** **OriginalFrame40Enable** **TestFrame40Enable**	**maxPort1Addresses** **Int** **port1Frame40Enable** **port1Frame40Enable**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPort1Addresses.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: port1Frame40Enable.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalFrame40Enable.	
4	IF OriginalFrame40Enable IS_EQUAL_TO 0.	
4.1	ASSIGN TestFrame40Enable EQUALS 1.	
4.2	GOTO step 6.	
5	ASSIGN TestFrame40Enable EQUALS 0.	
6	ASSIGN port1Frame40Enable.Table_Row EQUALS TestFrame40Enable.	
7	SET the following objects: port1Frame40Enable.Table_Row.	Pass/Fail
8	GET the following objects: port1Frame40Enable.Table_Row.	Pass/Fail
9	VERIFY port1Frame40Enable.Table_Row IS_EQUAL_TO TestFrame40Enable.	Pass/Fail
10	ASSIGN port1Frame40Enable.Table_Row EQUALS OriginalFrame40Enable.	
11	SET the following objects: port1Frame40Enable.Table_Row.	Pass/Fail
12	GET the following objects: port1Frame40Enable.Table_Row.	Pass/Fail
13	VERIFY port1Frame40Enable.Table_Row IS_EQUAL_TO OriginalFrame40Enable.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.13 Monitor TS2 Port 1 Status

Test Procedure:	**Monitor TS2 Port 1 Status**	
Description:	**This test case verifies that the ASC allows a management station to identify if a device is present for a TS2 Port 1 address.**	
Requirement(s):	• **3.5.2.2.9.1 Monitor TS2 Port 1 Status**	
Variable(s):	**MaxRows** **Table_Row**	**maxPort1Addresses** **Int**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPort1Addresses.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
3	GET the following objects: port1Frame40Enable.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.14 Monitor TS2 Port 1 Fault Frame

Test Procedure:	**Monitor TS2 Port 1 Fault Frame**	
Description:	**This test case verifies that the ASC allows a management station to identify if Frame 40 message to the device is enabled for a TS2 Port 1 address.**	
Requirement(s):	• **3.5.2.2.9.2 Monitor TS2 Port 1 Fault Frame**	
Variable(s):	**MaxRows** **Table_Row**	**maxTimebaseAscActions** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPort1Addresses.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
3	GET the following objects: port1Number.Table_Row, port1Frame40Enable.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.15 Monitor ATC Serial Bus 1 Status

Test Procedure:	**Monitor ATC Serial Bus 1 Status**

Description:	**This test case verifies that the ASC allows a management station to identify the communications status of devices with a Serial Bus 1 address.**	
Requirement(s):	• **3.5.2.2.9.3 Monitor ATC Serial Bus 1 Status**	
Variable(s):	**MaxRows** **Table_Row**	**maxSIUPort1Addresses** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSIUPort1Addresses.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
3	GET the following objects: siuPort1DevicePresent.Table_Row, siuPort1Status.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.16 Monitor Signal Monitoring Unit Channel Voltage

Test Procedure:	**Monitor Signal Monitoring Unit Channel Voltage**	
Description:	**This test case verifies that the ASC allows a management station to view voltage levels in the SMU.**	
Requirement(s):	• **3.5.2.2.10.1 Monitor Signal Monitoring Unit Channel Voltage**	
Variable(s):	**MaxChannels** **Table_Index1** **Table_Index2**	**maxChannels** **Int** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannels.	Pass/Fail
1.1	RECORD this information as MaxChannels.	
2	FOR Table_Index1 from 1 to MaxRows.	
2.1	FOR Table_Index1 from 1 to 3.	
2.1.1	GET the following objects: ascSmuVoltage.Table_Index1.Table_Index2	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.9.17 Monitor Signal Monitoring Unit Channel Current

Test Procedure:	**Monitor Signal Monitoring Unit Channel Current**
Description:	**This test case verifies that the ASC allows a management station to view current levels in the SMU.**
Requirement(s):	• **3.5.2.2.10.2 Monitor Signal Monitoring Unit Channel Current**
Variable(s):	**MaxChannels** **maxChannels** **Table_Index1** **Int** **Table_Index2** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannels.	Pass/Fail
1.1	RECORD this information as MaxChannels.	
2	FOR Table_Index1 from 1 to MaxRows.	
2.1	FOR Table_Index1 from 1 to 3.	
2.1.1	GET the following objects: ascSmuCurrent.Table_Index1.Table_Index2	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.9.18 Enable/Disable ECLA Communications

Test Procedure:	**Enable/Disable ECLA Communications**
Description:	**This test case verifies that the ASC allows a management station to enable or disable communication with an ECLA**
Requirement(s):	• **3.5.1.10.1.1 Enable ECLA Communications** • **3.5.1.10.1.2 Disable ECLA Communications**
Variable(s):	**OriginalSetting** **eclaCommEnable** **TestSetting** **eclaCommEnable**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.

Test Step Number	Test Procedure	Results
1	GET the following objects: eclaCommEnable	Pass/Fail
1.1	RECORD this information as OriginalSetting.	
2	IF OriginalSetting IS_EQUAL_TO 0.	
2.1	ASSIGN TestSetting EQUALS 1.	
2.2	GO step 4.	
3	ASSIGN TestOutputControl EQUALS 0.	
4	ASSIGN specialFunctionOutputControl.Table_Row EQUALS TestOutputControl.	
5	SET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
6	GET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
7	VERIFY specialFunctionOutputControl.Table_Row IS_EQUAL_TO TestOutputControl.	Pass/Fail
8	ASSIGN specialFunctionOutputControl.Table_Row EQUALS OriginalOutputControl.	
9	SET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
10	GET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
11	VERIFY specialFunctionOutputControl.Table_Row IS_EQUAL_TO OriginalOutputControl.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10 Compressed Block Objects

C.3.10.1 Configure Phase Data Block

Test Procedure:	**Configure Phase Data Block**
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure phase sets. If the ASC does not support multiple phase sets, it configures the phase table instead of the phase set table.**
Requirement(s):	• **3.5.2.1.14.1.1 Configure Block Object Get Control - Phase Data** • **3.5.2.1.14.2 Monitor Block Error – Error-causing Data Element**
Variable(s):	**OriginalBlock** **ascBlockData** **TestBlock** **ascBlockData** **MaxPhases** **maxPhases** **Phase** **Octet String**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test procedure step 11.1.1 is executed because the SET function in Step 11 resulted in an error.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of phases supported by the ASC'.	
1.1	RECORD this information as MaxPhases.	
2	ASSIGN Phase EQUALS RANDOM (1 TO MaxPhases).	
3	ASSIGN ascBlockGetControl EQUALS '00 00 Phase 01 01 01'.	
3.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
3.2	NOTE 'ascBlockDataID = 0 (phase data)'.	
3.3	NOTE 'ascBlockIndex1 = Phase in hexadecimal format'.	
3.4	NOTE 'ascBlockQuantity1 = 1 (number of phases = 1)'.	
3.5	NOTE 'ascBlockIndex2 = 1 (phase set = 1)'.	
3.6	NOTE 'ascBlockQuantity2 = 1 (number of phase sets = 1)'.	
4	SET the following objects: ascBlockGetControl.	Pass/Fail
5	GET the following objects: ascBlockErrorStatus.	Pass/Fail
6	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
7	GET the following objects: ascBlockData.	Pass/Fail
7.1	RECORD this information as OriginalBlock.	
8	USER-ACTION 'Generate an octet string with the modified valid values for AscPhaseBlockData. Valid values must pass a consistency check.'	
8.1	RECORD this information as TestBlock.	
9	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
10	ASSIGN ascBlockData EQUALS TestBlock.	
11	SET the following objects: ascBlockData.	Pass/Fail
11.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
11.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
11.1.2	EXIT	

12	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
13	GET the following objects: ascBlockData.	Pass/Fail
14	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
16	ASSIGN ascBlockData EQUALS OriginalBlock.	
17	SET the following objects: ascBlockData.	Pass/Fail
18	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
19	GET the following objects: ascBlockData.	Pass/Fail
20	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.2 Configure Vehicle Detector Data Block

Test Procedure:	**Configure Vehicle Detector Data Block**	
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure vehicle detector sets. If the ASC does not support multiple vehicle detector sets, it configures the vehicle detector table instead of the vehicle detector set table.**	
Requirement(s):	• **3.5.2.1.14.1.2 Configure Block Object Get Control – Vehicle Detector Data**	
Variable(s):	**OriginalBlock** **ascBlockData** **TestBlock** **ascBlockData** **MaxVehicleDetectors** **maxVehicleDetectors** **VehicleDetector** **Octet String**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test procedure step 12.1.1 is executed because the SET function in Step 12 resulted in an error.**	
Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'The ASC supports vehicle detectors'.	
2	PRE-CONDITION 'The user shall know the maximum number of vehicle detectors supported by the ASC'.	
2.1	RECORD this information as MaxVehicleDetectors.	
3	ASSIGN VehicleDetector EQUALS RANDOM (1 TO MaxVehicleDetectors).	
4	ASSIGN ascBlockGetControl EQUALS '00 01 VehicleDetector 01 01 01'.	
4.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
4.2	NOTE 'ascBlockDataID = 1 (vehicle detector data)'.	
4.3	NOTE 'ascBlockIndex1 = VehicleDetector in hexadecimal format'.	

4.4	NOTE 'ascBlockQuantity1 = 1 (number of vehicle detectors = 1)'.	
4.5	NOTE 'ascBlockIndex2 = 1 (vehicle detector set = 1)'.	
4.6	NOTE 'ascBlockQuantity2 = 1 (number of vehicle detector sets = 1)'.	
5	SET the following objects: ascBlockGetControl.	Pass/Fail
6	GET the following objects: ascBlockErrorStatus.	Pass/Fail
7	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
8	GET the following objects: ascBlockData.	Pass/Fail
8.1	RECORD this information as OriginalBlock.	
9	USER-ACTION 'Generate an octet string with the modified valid values for AscVehDetectorBlock. Valid values must pass a consistency check.'	
9.1	RECORD this information as TestBlock.	
10	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
11	ASSIGN ascBlockData EQUALS TestBlock.	
12	SET the following objects: ascBlockData.	Pass/Fail
12.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
12.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
12.1.2	EXIT	
13	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
14	GET the following objects: ascBlockData.	Pass/Fail
15	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
17	ASSIGN ascBlockData EQUALS OriginalBlock.	
18	SET the following objects: ascBlockData.	Pass/Fail
19	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
20	GET the following objects: ascBlockData.	Pass/Fail
21	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.3 Configure Pedestrian Detector Data Block

Test Procedure:	**Configure Pedestrian Detector Data Block**	
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure pedestrian detectors.**	
Requirement(s):	• **3.5.2.1.14.1.3 Configure Block Object Get Control – Pedestrian Detector Data**	
Variable(s):	**OriginalBlock** **TestBlock** **MaxPedestrianDetectors** **PedestrianDetector**	**ascBlockData** **ascBlockData** **maxPedestrianDetectors** **Octet String**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test procedure step 12.1.1 is executed because the SET function in Step 12 resulted in an error.

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The ASC supports pedestrian detectors'.	
2	PRE-CONDITION 'The user shall know the maximum number of pedestrian detectors supported by the ASC'.	
2.1	RECORD this information as MaxPedestrianDetectors.	
3	ASSIGN PedestrianDetector EQUALS RANDOM (1 TO MaxPedestrianDetectors).	
4	ASSIGN ascBlockGetControl EQUALS '00 02 PedestrianDetector 01'.	
4.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
4.2	NOTE 'ascBlockDataID = 2 (pedestrian detector data)'.	
4.3	NOTE 'ascBlockIndex1 = PedestrianDetector in hexadecimal format'.	
4.4	NOTE 'ascBlockQuantity1 = 1 (number of pedestrian detectors = 1)'.	
5	SET the following objects: ascBlockGetControl.	Pass/Fail
6	GET the following objects: ascBlockErrorStatus.	Pass/Fail
7	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
8	GET the following objects: ascBlockData.	Pass/Fail
8.1	RECORD this information as OriginalBlock.	
9	USER-ACTION 'Generate an octet string with the modified valid values for AscPedDetectorBlock. Valid values must pass a consistency check.'	
9.1	RECORD this information as TestBlock.	
10	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
11	ASSIGN ascBlockData EQUALS TestBlock.	
12	SET the following objects: ascBlockData.	Pass/Fail
12.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
12.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
12.1.2	EXIT	
13	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
14	GET the following objects: ascBlockData.	Pass/Fail
15	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
17	ASSIGN ascBlockData EQUALS OriginalBlock.	
18	SET the following objects: ascBlockData.	Pass/Fail
19	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
20	GET the following objects: ascBlockData.	Pass/Fail
21	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.4 Configure Pattern Data Block

Test Procedure:	**Configure Pattern Data Block**
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure timing patterns.**
Requirement(s):	• **3.5.2.1.14.1.4 Configure Block Object Get Control – Pattern Data**
Variable(s):	**OriginalBlock ascBlockData** **TestBlock ascBlockData** **MaxPatterns maxPatterns** **Pattern Octet String**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test procedure step 12.1.1 is executed because the SET function in Step 12 resulted in an error.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The ASC supports timing patterns'.	
2	PRE-CONDITION 'The user shall know the maximum number of patterns supported by the ASC'.	
2.1	RECORD this information as MaxPatterns.	
3	ASSIGN Pattern EQUALS RANDOM (1 TO MaxPatterns).	
4	ASSIGN ascBlockGetControl EQUALS '00 03 Pattern 01'.	
4.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
4.2	NOTE 'ascBlockDataID = 3 (pattern data)'.	
4.3	NOTE 'ascBlockIndex1 = Pattern in hexadecimal format'.	
4.4	NOTE 'ascBlockQuantity1 = 1 (number of patterns = 1)'.	
5	SET the following objects: ascBlockGetControl.	Pass/Fail
6	GET the following objects: ascBlockErrorStatus.	Pass/Fail
7	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
8	GET the following objects: ascBlockData.	Pass/Fail
8.1	RECORD this information as OriginalBlock.	
9	USER-ACTION 'Generate an octet string with the modified valid values for AscPatternBlock. Valid values must pass a consistency check.'	
9.1	RECORD this information as TestBlock.	
10	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
11	ASSIGN ascBlockData EQUALS TestBlock.	
12	SET the following objects: ascBlockData.	Pass/Fail
12.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
12.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
12.1.2	EXIT	
13	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
14	GET the following objects: ascBlockData.	Pass/Fail
15	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	

17	ASSIGN ascBlockData EQUALS OriginalBlock.	
18	SET the following objects: ascBlockData.	Pass/Fail
19	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
20	GET the following objects: ascBlockData.	Pass/Fail
21	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.5 Configure Split Data Block

Test Procedure:	**Configure Split Data Block**	
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure splits.**	
Requirement(s):	• **3.5.2.1.14.1.5 Configure Block Object Get Control – Split Data**	
Variable(s):	**OriginalBlock** **ascBlockData** **TestBlock** **ascBlockData** **MaxPhases** **maxPhases** **MaxSplits** **maxSplits** **Phase** **Octet String** **Split** **Octet String**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test procedure step 14.1.1 is executed because the SET function in Step 14 resulted in an error.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The ASC supports splits'.	
2	PRE-CONDITION 'The user shall know the maximum number of patterns supported by the ASC'.	
2.1	RECORD this information as MaxPatterns.	
3	PRE-CONDITION 'The user shall know the maximum number of splits supported by the ASC'.	
3.1	RECORD this information as MaxSplits.	
4	ASSIGN Pattern EQUALS RANDOM (1 TO MaxPatterns).	
5	ASSIGN Split EQUALS RANDOM (1 TO MaxSplits).	
6	ASSIGN ascBlockGetControl EQUALS '00 04 Phase 01 Split 01'.	
6.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
6.2	NOTE 'ascBlockDataID = 4 (split data)'.	
6.3	NOTE 'ascBlockIndex1 = Phase in hexadecimal format'.	
6.4	NOTE 'ascBlockQuantity1 = 1 (number of phases = 1)'.	
6.5	NOTE 'ascBlockIndex2 = Split in hexadecimal format'.	
6.6	NOTE 'ascBlockQuantity2 = 1 (number of splits = 1)'.	

7	SET the following objects: ascBlockGetControl.	Pass/Fail
8	GET the following objects: ascBlockErrorStatus.	Pass/Fail
9	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
10	GET the following objects: ascBlockData.	Pass/Fail
10.1	RECORD this information as OriginalBlock.	
11	USER-ACTION 'Generate an octet string with the modified valid values for AscSplitBlock. Valid values must pass a consistency check.'	
11.1	RECORD this information as TestBlock.	
12	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
13	ASSIGN ascBlockData EQUALS TestBlock.	
14	SET the following objects: ascBlockData.	Pass/Fail
14.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
14.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
14.1.2	EXIT	
16	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
17	GET the following objects: ascBlockData.	Pass/Fail
18	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
19	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
20	ASSIGN ascBlockData EQUALS OriginalBlock.	
21	SET the following objects: ascBlockData.	Pass/Fail
22	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
23	GET the following objects: ascBlockData.	Pass/Fail
24	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.6 Configure Overlap Data Block

Test Procedure:	**Configure Overlap Data Block**
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure overlap sets. If the ASC does not support multiple overlap detector sets, it configures the overlap table instead of the overlap set table.**
Requirement(s):	• **3.5.2.1.14.1.6 Configure Block Object Get Control – Overlap Data**
Variable(s):	**OriginalBlock ascBlockData** **TestBlock ascBlockData** **MaxOverlaps maxOverlaps** **Overlap Octet String**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test**

	procedure step 12.1.1 is executed because the SET function in Step 12 resulted in an error.

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The ASC supports overlaps'.	
2	PRE-CONDITION 'The user shall know the maximum number of overlaps supported by the ASC'.	
2.1	RECORD this information as MaxOverlaps.	
3	ASSIGN Overlap EQUALS RANDOM (1 TO MaxOverlaps).	
4	ASSIGN ascBlockGetControl EQUALS '00 06 Overlap 01 01 01'.	
4.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
4.2	NOTE 'ascBlockDataID = 5 (overlap data)'.	
4.3	NOTE 'ascBlockIndex1 = Overlap in hexadecimal format'.	
4.4	NOTE 'ascBlockQuantity1 = 1 (number of overlaps = 1)'.	
4.5	NOTE 'ascBlockIndex2 = 1 (overlap set = 1)'.	
4.6	NOTE 'ascBlockQuantity2 = 1 (number of overlap sets = 1)'.	
5	SET the following objects: ascBlockGetControl.	Pass/Fail
6	GET the following objects: ascBlockErrorStatus.	Pass/Fail
7	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
8	GET the following objects: ascBlockData.	Pass/Fail
8.1	RECORD this information as OriginalBlock.	
9	USER-ACTION 'Generate an octet string with the modified valid values for AscOverlapBlock. Valid values must pass a consistency check.'	
9.1	RECORD this information as TestBlock.	
10	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
11	ASSIGN ascBlockData EQUALS TestBlock.	
12	SET the following objects: ascBlockData.	Pass/Fail
12.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
12.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
12.1.2	EXIT	
13	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
14	GET the following objects: ascBlockData.	Pass/Fail
15	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
16	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
17	ASSIGN ascBlockData EQUALS OriginalBlock.	
18	SET the following objects: ascBlockData.	Pass/Fail
19	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
20	GET the following objects: ascBlockData.	Pass/Fail
21	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.7 Configure Preempt Data Block

Test Procedure:	**Configure Preempt Data Block**
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure preempts.**
Requirement(s):	• **3.5.2.1.14.1.7 Configure Block Object Get Control - Preempt Data**
Variable(s):	**OriginalBlock ascBlockData** **TestBlock ascBlockData** **MaxPreempts maxPreempts** **Preempt Octet String**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test procedure step 12.1.1 is executed because the SET function in Step 12 resulted in an error.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The ASC supports preempts'.	
2	PRE-CONDITION 'The user shall know the maximum number of preempts supported by the ASC'.	
2.1	RECORD this information as MaxPreempts.	
3	ASSIGN Preempt EQUALS RANDOM (1 TO MaxPreempts).	
4	ASSIGN ascBlockGetControl EQUALS '00 06 Preempt 01'.	
4.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
4.2	NOTE 'ascBlockDataID = 6 (preempt data)'.	
4.3	NOTE 'ascBlockIndex1 = Preempt in hexadecimal format'.	
4.4	NOTE 'ascBlockQuantity1 = 1 (number of preempts = 1)'.	
5	SET the following objects: ascBlockGetControl.	Pass/Fail
6	GET the following objects: ascBlockErrorStatus.	Pass/Fail
7	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
8	GET the following objects: ascBlockData.	Pass/Fail
8.1	RECORD this information as OriginalBlock.	
9	USER-ACTION 'Generate an octet string with the modified valid values for AscPreemptBlock. Valid values must pass a consistency check.'	
9.1	RECORD this information as TestBlock.	
10	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
11	ASSIGN ascBlockData EQUALS TestBlock.	
12	SET the following objects: ascBlockData.	Pass/Fail
12.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
12.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
12.1.2	EXIT	
13	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
14	GET the following objects: ascBlockData.	Pass/Fail
15	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail

16	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
17	ASSIGN ascBlockData EQUALS OriginalBlock.	
18	SET the following objects: ascBlockData.	Pass/Fail
19	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
20	GET the following objects: ascBlockData.	Pass/Fail
21	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.8 Configure Sequence Data Block

Test Procedure:	**Configure Sequence Data Block**	
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure phase sequence data.**	
Requirement(s):	• **3.5.2.1.14.1.8 Configure Block Object Get Control – Sequence**	
Variable(s):	**OriginalBlock** **ascBlockData** **TestBlock** **ascBlockData** **MaxSequences** **maxSequences** **MaxRings** **maxRings** **Ring** **Octet String** **Sequence** **Octet String**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test procedure step 13.1.1 is executed because the SET function in Step 13 resulted in an error.**	
Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'The user shall know the maximum number of sequences supported by the ASC'.	
1.1	RECORD this information as MaxSequences.	
2	PRE-CONDITION 'The user shall know the maximum number of rings supported by the ASC'.	
2.1	RECORD this information as MaxRings.	
3	ASSIGN Ring EQUALS RANDOM (1 TO MaxRings).	
4	ASSIGN Sequence EQUALS RANDOM (1 TO MaxSequence).	
5	ASSIGN ascBlockGetControl EQUALS '00 07 Ring 01 Sequence 01'.	
5.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
5.2	NOTE 'ascBlockDataID = 7 (sequence data)'.	
5.3	NOTE 'ascBlockIndex1 = Ring in hexadecimal format'.	
5.4	NOTE 'ascBlockQuantity1 = 1 (number of rings = 1)'.	
5.5	NOTE 'ascBlockIndex2 = Sequence in hexadecimal format'.	

5.6	NOTE 'ascBlockQuantity2 = 1 (number of sequences = 1)'.	
6	SET the following objects: ascBlockGetControl.	Pass/Fail
7	GET the following objects: ascBlockErrorStatus.	Pass/Fail
8	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
9	GET the following objects: ascBlockData.	Pass/Fail
9.1	RECORD this information as OriginalBlock.	
10	USER-ACTION 'Generate an octet string with the modified valid values for AscSequenceBlock. Valid values must pass a consistency check.'	
10.1	RECORD this information as TestBlock.	
11	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
12	ASSIGN ascBlockData EQUALS TestBlock.	
13	SET the following objects: ascBlockData.	Pass/Fail
13.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
13.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
13.1.2	EXIT	
14	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
15	GET the following objects: ascBlockData.	Pass/Fail
16	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
17	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
18	ASSIGN ascBlockData EQUALS OriginalBlock.	
19	SET the following objects: ascBlockData.	Pass/Fail
20	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
21	GET the following objects: ascBlockData.	Pass/Fail
22	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.10.9 Configure Channel Data Block

Test Procedure:	**Configure Channel Data Block**
Description:	**This test case verifies that the ASC allows a management station to use the block object to configure channels.**
Requirement(s):	• **3.5.2.1.14.1.9 Configure Block Object Get Control – Channels Data**
Variable(s):	**OriginalBlock ascBlockData** **TestBlock ascBlockData** **MaxChannels maxChannels** **Channel Octer String**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case. This case fails if the test**

	procedure step 11.1.1 is executed because the SET function in Step 11 resulted in an error.

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of channels supported by the ASC'.	
1.1	RECORD this information as MaxChannels.	
2	ASSIGN Channel EQUALS RANDOM (1 TO MaxChannel).	
3	ASSIGN ascBlockGetControl EQUALS '00 08 Channel 01'.	
3.1	NOTE 'ascBlockDataType = 0 (standard block)'.	
3.2	NOTE 'ascBlockDataID = 8 (channel data)'.	
3.3	NOTE 'ascBlockIndex1 = Channel in hexadecimal format'.	
3.4	NOTE 'ascBlockQuantity1 = 1 (number of channels = 1)'.	
4	SET the following objects: ascBlockGetControl.	Pass/Fail
5	GET the following objects: ascBlockErrorStatus.	Pass/Fail
6	VERIFY ascBlockErrorStatus IS_EQUAL_TO 0.	Pass/Fail
7	GET the following objects: ascBlockData.	Pass/Fail
7.1	RECORD this information as OriginalBlock.	
8	USER-ACTION 'Generate an octet string with the modified valid values for AscChannelBlock. Valid values must pass a consistency check.'	
8.1	RECORD this information as TestBlock.	
9	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
10	ASSIGN ascBlockData EQUALS TestBlock.	
11	SET the following objects: ascBlockData.	Pass/Fail
11.1	IF ERROR INDEX IS_NOT_EQUAL_TO 0.	
11.1.1	GET the following objects: ascBlockErrorStatus.	Pass/Fail
11.1.2	EXIT	
12	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
13	GET the following objects: ascBlockData.	Pass/Fail
14	VERIFY ascBlockData IS_EQUAL_TO TestBlock.	Pass/Fail
15	PERFORM the Test Procedure 'C.3.1.1 Create Database Transaction'.	
16	ASSIGN ascBlockData EQUALS OriginalBlock.	
17	SET the following objects: ascBlockData.	Pass/Fail
18	PERFORM the Test Procedure 'C.3.1.2 Verify Database Status and Verify Database Error'.	
19	GET the following objects: ascBlockData.	Pass/Fail
20	VERIFY ascBlockData IS_EQUAL_TO OriginalBlock.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11 Monitor Signal Operations

C.3.11.1 Monitor External Alarm States

Test Procedure:	**Monitor External Alarm States**	
Description:	**This test case verifies that the ASC allows a management station to view active state of all user configured alarm inputs.**	
Requirement(s):	• **3.5.2.2.1.1 Monitor External Alarm States**	
Variable(s):	**MaxRows** **Table_Row**	**maxAlarmGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxAlarmGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: alarmGroupState.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.2 Monitor Unit Level Alarms - Short

Test Procedure:	**Monitor Unit Level Alarms - Short**
Description:	**This test case verifies that the ASC allows a management this test case verifies that the ASC allows to view alarms if any of the following conditions are active:** • **External Alarm Active** • **Preempt Active** • **Flash Active** • **Offset Transitioning** • **Local Override** • **Coordination Alarm** • **Detector Fault** • **Stop Time Active** • **Priority Call Active**
Requirement(s):	• **3.5.2.2.1.2 Monitor External Alarm Active** • **3.5.2.2.1.3 Monitor Flash Active** • **3.5.2.2.1.4 Monitor Local Override** • **3.5.2.2.1.5 Monitor Coordination Alarm** • **3.5.2.2.1.6 Monitor Detector Fault**

	• **3.5.2.2.1.7 Monitor Stop Time Active** • **3.5.2.2.2.2 Monitor Preempt Active** • **3.5.2.2.2.3 Monitor Offset Transitioning** • **3.5.2.2.2.4 Monitor Priority Call Active**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: shortAlarmStatusV4.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.3 Monitor Unit Level Alarms

Test Procedure:	**Monitor Unit Level Alarms**
Description:	**This test case verifies that the ASC allows a management this test case verifies that the ASC allows to view alarms if any of the following conditions are active:** • **Cycle Fault** • **Coordination Fault** • **Coordination Fail** • **Cycle Fail** • **Free Mode** • **Coordination Active** • **Cabinet IO Link Error** • **SMU Communications Error** • **Preempt Maximum Presence** • **ECLA Active** • **RSU Interface Watchdog** • **CV Certificate Faults**
Requirement(s):	• **3.5.2.2.1.8 Monitor Cycle Fault** • **3.5.2.2.1.9 Monitor Coordination Fault** • **3.5.2.2.1.10 Monitor Coordination Fail Alarm** • **3.5.2.2.1.11 Monitor Cycle Fail Alarm** • **3.5.2.2.1.12 Monitor Cabinet IO Link Alarm** • **3.5.2.2.1.13 Monitor SMU Communications Error** • **3.5.2.2.1.14 Monitor Preempt Maximum Presence Alarm** • **3.5.2.2.2.5 Monitor Local Free Status** • **3.5.2.2.2.6 Monitor Coordination Active** • **3.5.2.2.2.7 Monitor ECLA Active** • **3.5.4.1.6 Monitor Interface Watchdog Alarm** • **3.5.4.3.4 Monitor CV Certificate Fauts**
Variable(s):	

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: unitAlarmStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.4 Monitor Flash Status

Test Procedure:	**Monitor Flash Status**
Description:	**This test case verifies that the ASC allows a management station to identify why the ASC is in Flash.**
Requirement(s):	• **3.5.2.2.1.3 Monitor Flash Status**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	PERFORM the Test Procedure 'C.3.11.2 Monitor Unit Level Alarms - Short'.	
2	GET the following objects: unitFlashStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.5 Monitor Unit Control Status

Test Procedure:	**Monitor Unit Control Status**
Description:	**This test case verifies that the ASC allows a management station to monitor the control mode for pattern, flash, or free.**
Requirement(s):	• **3.5.2.2.2.1 Monitor Unit Control Status**

Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: unitControlStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.6 Monitor Local Free Status

Test Procedure:	**Monitor Local Free Status**	
Description:	**This test case verifies that the ASC allows a management station to identify why an ASC is running in Free Mode.**	
Requirement(s):	• **3.5.2.2.2.5 Monitor Local Free Status**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	PERFORM the Test Procedure 'C.3.11.3 Monitor Unit Level Alarms'.	
2	GET the following objects: localFreeStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.7 Monitor Current Pattern Status

Test Procedure:	**Monitor Current Pattern Status**
Description:	**This test case verifies that the ASC allows a management station to identify the current coordination pattern/mode**
Requirement(s):	• **3.5.2.2.2.8.1 Monitor Current Pattern Status**

Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: coordPatternStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.8 Monitor Current Pattern Command Source

Test Procedure:	**Monitor Current Pattern Command Source**	
Description:	**This test case verifies that the ASC allows a management station to identify why the currently active pattern is active.**	
Requirement(s):	• **3.5.2.2.2.8.2 Monitor Current Pattern Command Source**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: coordPatternSource.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.9 Monitor Current Pattern Fault Status

Test Procedure:	**Monitor Current Pattern Fault Status**	
Description:	**This test case verifies that the ASC allows a management station to identify why the current pattern is invalid if it is invalid.**	
Requirement(s):	• **3.5.2.2.2.8.3 Monitor Current Pattern Fault Status**	

Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: coordPatternFaultStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.10 Monitor Coordination Cycle Status

Test Procedure:	**Monitor Coordination Cycle Status**
Description:	**This test case verifies that the ASC allows a management station to identify how many seconds are remaining in the local cycle.**
Requirement(s):	• **3.5.2.2.2.9.1 Monitor Coordination Cycle Status**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: coordCycleStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.11 Monitor Coordination Synchronization Status

Test Procedure:	**Monitor Coordination Synchronization Status**
Description:	**This test case verifies that the ASC allows a management station to identify the current position of the master cycle in seconds.**
Requirement(s):	• **3.5.2.2.2.9.2 Monitor Coordination Synchronization Status**

Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: coordSyncStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.12 Monitor Current Offset

Test Procedure:	**Monitor Coordination Synchronization Status**	
Description:	**This test case verifies that the ASC allows a management station to identify the current coordination offset in effect.**	
Requirement(s):	• **3.5.2.2.2.9.3 Monitor Current Offset**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: coordCurrentOffset.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.13 Monitor Active Red Phases

Test Procedure:	**Monitor Active Red Phases**	
Description:	**This test case verifies that the ASC allows a management station to identify the phases with active RED indications.**	
Requirement(s):	• **3.5.2.2.3.1 Monitor Active Red Phases**	

Variable(s):	**MaxRows** **maxPhaseGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupReds.Table_Row.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.14 Monitor Active Yellow Phases

Test Procedure:	**Monitor Active Yellow Phases**
Description:	**This test case verifies that the ASC allows a management station to identify the phases with active YELLOW indications.**
Requirement(s):	• **3.5.2.2.3.2 Monitor Active Yellow Phases**
Variable(s):	**MaxRows** **maxPhaseGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupYellows.Table_Row.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.15 Monitor Active Green Phases

Test Procedure:	**Monitor Active Green Phases**	
Description:	**This test case verifies that the ASC allows a management station to identify the phases with active GREEN indications.**	
Requirement(s):	• **3.5.2.2.3.3 Monitor Active Green Phases**	
Variable(s):	**MaxRows** **Table_Row**	**maxPhaseGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupGreens.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.16 Monitor Active Don't Walk Phases

Test Procedure:	**Monitor Active Don't Walk Phases**	
Description:	**This test case verifies that the ASC allows a management station to identify the phases with active DON'T WALK indications.**	
Requirement(s):	• **3.5.2.2.3.4 Monitor Active Don't Walk Phases**	
Variable(s):	**MaxRows** **Table_Row**	**maxPhaseGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**

1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupDontWalks.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.17 Monitor Active Pedestrian Clearance Phases

Test Procedure:	**Monitor Active Pedestrian Clearance Phases**	
Description:	**This test case verifies that the ASC allows a management station to identify the phases with active flashing DON'T WALK indications.**	
Requirement(s):	• **3.5.2.2.3.5 Monitor Active Pedestrian Clearance Phases**	
Variable(s):	**MaxRows** **maxPhaseGroups** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupPedClears.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.18 Monitor Active Walk Phases

Test Procedure:	**Monitor Active Walk Phases**	
Description:	**This test case verifies that the ASC allows a management station to identify the phases with active WALK indications.**	
Requirement(s):	• **3.5.2.2.3.6 Monitor Active Walk Phases**	
Variable(s):	**MaxRows** **Table_Row**	**maxPhaseGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupWalks.Table_Row.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.19 Monitor Active On Phases

Test Procedure:	**Monitor Active On Phases**	
Description:	**This test case verifies that the ASC allows a management station to identify the phases that are currently active.**	
Requirement(s):	• **3.5.2.2.3.7 Monitor Active On Phases**	
Variable(s):	**MaxRows** **Table_Row**	**maxPhaseGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhaseGroups.	Pass/Fail

1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupPhaseOns.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.20 Monitor Next Phases

Test Procedure:	**Monitor Next Phases**	
Description:	**This test case verifies that the ASC allows a management station to identify the phases that are currently committed to be active next.**	
Requirement(s):	• **3.5.2.2.3.8 Monitor Next Phases**	
Variable(s):	**MaxRows** **Table_Row**	**maxPhaseGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: phaseStatusGroupPhaseOns.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.21 Monitor Phase Vehicle Calls

Test Procedure:	**Monitor Phase Vehicle Calls**
Description:	**This test case verifies that the ASC allows a management station to identify the phases currently have calls for vehicle, bicycle, or transit service placed on them.**

Requirement(s):	• **3.5.2.2.3.9 Monitor Phase Vehicle Calls**		
Variable(s):	**MaxRows** **Table_Row**	**maxPhaseGroups** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: maxPhaseGroups.		Pass/Fail
1.1	RECORD this information as MaxRows.		
2	FOR Table_Row from 1 to MaxRows.		
2.1	GET the following objects: phaseStatusGroupVehCalls.Table_Row.		Pass/Fail
Test Procedure Results			
Tested By:		**Date Tested:**	**Pass/Fail**
Test Procedure Notes:			

C.3.11.22 Monitor Phase Pedestrian Calls

Test Procedure:	**Monitor Phase Vehicle Calls**		
Description:	**This test case verifies that the ASC allows a management station to identify the phases currently have calls for pedestrian service placed on them.**		
Requirement(s):	• **3.5.2.2.3.9 Monitor Phase Vehicle Calls**		
Variable(s):	**MaxRows** **Table_Row**	**maxPhaseGroups** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: maxPhaseGroups.		Pass/Fail
1.1	RECORD this information as MaxRows.		
2	FOR Table_Row from 1 to MaxRows.		
2.1	GET the following objects: phaseStatusGroupPedCalls.Table_Row.		Pass/Fail
Test Procedure Results			

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.11.23 Monitor Ring Status

Test Procedure:	**Monitor Ring Status**
Description:	**This test case verifies that the ASC allows a management station to identify the status of each ring in the currently active sequence.**
Requirement(s):	• **3.5.2.2.4.1 Monitor Ring Status** • **3.5.2.2.4.2 Monitor Ring Termination Cause**
Variable(s):	**MaxRows** **maxRings** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRings.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: ringStatus.Table_Row.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.24 Monitor Current Phase On Time

Test Procedure:	**Monitor Current Phase On Time**
Description:	**This test case verifies that the ASC allows a management station to identify how the time each currently active phase has been on for.**
Requirement(s):	• **3.5.2.2.4.3 Monitor Current Phase On Time**
Variable(s):	**MaxRows** **maxRings** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRings.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: ringCurrentPhase.Table_Row, ringCurrentOnTime.Table_Row.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.25 Monitor Active Red Channels

Test Procedure:	**Monitor Active Red Channels**
Description:	**This test case verifies that the ASC allows a management station to identify channels currently outputting a RED indication.**
Requirement(s):	• **3.5.2.2.5.1 Monitor Active Red Channels**
Variable(s):	**MaxRows** **maxChannelStatusGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannelStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: channelStatusGroupReds.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.26 Monitor Active Yellow Channels

Test Procedure:	**Monitor Active Yellow Channels**

Description:	**This test case verifies that the ASC allows a management station to identify channels currently outputting a YELLOW indication.**
Requirement(s):	• **3.5.2.2.5.2 Monitor Active Yellow Channels**
Variable(s):	**MaxRows** **maxChannelStatusGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannelStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: channelStatusGroupYellows.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.27 Monitor Active Green Channels

Test Procedure:	**Monitor Active Green Channels**
Description:	**This test case verifies that the ASC allows a management station to identify channels currently outputting a GREEN indication.**
Requirement(s):	• **3.5.2.2.5.3 Monitor Active Green Channels**
Variable(s):	**MaxRows** **maxChannelStatusGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxChannelStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: channelStatusGroupGreens.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.28 Monitor Active Red Overlaps

Test Procedure:	**Monitor Active Red Overlaps**	
Description:	**This test case verifies that the ASC allows a management station to identify the overlaps with active RED indications.**	
Requirement(s):	• **3.5.2.2.6.1 Monitor Active Red Overlaps**	
Variable(s):	**MaxRows** **Table_Row**	**maxOverlapStatusGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlapStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: overlapStatusGroupReds.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.29 Monitor Active Yellow Overlaps

Test Procedure:	**Monitor Active Yellow Overlaps**	
Description:	**This test case verifies that the ASC allows a management station to identify the overlaps with active YELLOW indications.**	
Requirement(s):	• **3.5.2.2.6.2 Monitor Active Red Overlaps**	
Variable(s):	**MaxRows** **Table_Row**	**maxOverlapStatusGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlapStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	

2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: overlapStatusGroupYellows.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.30 Monitor Active Green Overlaps

Test Procedure:	**Monitor Active Green Overlaps**	
Description:	**This test case verifies that the ASC allows a management station to identify the overlaps with active GREEN indications.**	
Requirement(s):	• **3.5.2.2.6.3 Monitor Active Green Overlaps**	
Variable(s):	**MaxRows** **Table_Row**	**maxOverlapStatusGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxOverlapStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: overlapStatusGroupGreens.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.31 Monitor Active Overlap Flashing Yellow Arrows

Test Procedure:	**Monitor Active Overlap Flashing Yellow Arrow**	
Description:	**This test case verifies that the ASC allows a management station to identify active Flashing YELLOW Arrow indications.**	
Requirement(s):	• **3.5.2.2.6.4 Monitor Active Overlap Flashing Yellow Arrows**	
Variable(s):	**MaxRows** **Table_Row**	**maxOverlapStatusGroups** **Int**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxOverlapStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	IF FYA IS NOT WIRED	
2.1	GOTO step 4.	
3	FOR Table_Row from 1 to MaxRows.	
3.1	GET the following objects: overlapStatusGroupYellows.Table_Row.	Pass/Fail
3.2	EXIT	
4	FOR Table_Row from 1 to MaxRows.	
4.1	GET the following objects: overlapStatusGroupGreens.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.32 Monitor Active Overlap Flashing Red Arrows

Test Procedure:	**Monitor Active Overlap Flashing Red Arrow**	
Description:	**This test case verifies that the ASC allows a management station to identify active Flashing RED Arrow indications.**	
Requirement(s):	• **3.5.2.2.6.5 Monitor Active Overlap Flashing Yellow Arrows**	
Variable(s):	**MaxRows** **maxOverlapStatusGroups** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxOverlapStatusGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	IF FYA IS NOT WIRED	
2.1	GOTO step 4.	
3	FOR Table_Row from 1 to MaxRows.	
3.1	GET the following objects: overlapStatusGroupReds.Table_Row.	Pass/Fail
3.2	EXIT	
4	FOR Table_Row from 1 to MaxRows.	

4.1	GET the following objects: overlapStatusGroupGreens.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.33 Monitor Currently Active Preempt

Test Procedure:	**Monitor Currently Active Preempt**
Description:	**This test case verifies that the ASC allows a management station to view the currently active preempt.**
Requirement(s):	• **3.5.2.2.7.1 Monitor Currently Active Preempt**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: preemptStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.34 Monitor Currently Preempt Inputs

Test Procedure:	**Monitor Current Preempt Inputs**	
Description:	**This test case verifies that the ASC allows a management station to view the input state for each preempt.**	
Requirement(s):	• **3.5.2.2.7.2 Monitor Current Preempt Inputs**	
Variable(s):	**MaxRows** **Table_Row**	**maxPreemptGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreemptGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: preemptStatusGroup.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.35 Monitor Currently Preempt State

Test Procedure:	**7.39 Monitor Current Preempt State**
Description:	**This test case verifies that the ASC allows a management station to identify the preempt status of the current active preempt.**
Requirement(s):	• **3.5.2.2.7.3 Monitor Current Preempt State**
Variable(s):	**MaxRows** **maxPreempts** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: preemptState.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.36 Monitor Current Gate Status

Test Procedure:	**Monitor Current Gate Status**
Description:	**This test case verifies that the ASC allows a management station to view the status and description of each gate.**
	• **3.5.2.2.7.4 Monitor Current Gate Status**

Requirement(s):		
Variable(s):	**MaxRows** **Table_Row**	**maxPreemptGates** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreemptGates.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1.1	GET the following objects: preemptGateStatus.Table_Row, preemptGateDescription.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.37 Determine Maximum Number of Special Functions

Test Procedure:	**Determine Maximum Number of Special Functions**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of special functions supported by ASC and verifies that the ASC supports the minimum number of special functions required by the user.**	
Requirement(s):	• **3.5.2.2.8.1 Determine Maximum Number of Special Functions**	
Variable(s):	**UserMinSpecialFunctionOutputs**	**maxSpecialFunctionOutputs**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	USER-ACTION 'Determine the number of special functions required as specified in FR ID 3.5.2.2.8.1 of the PRL.'	
1.1	RECORD this information as UserMinSpecialFunctionOutputs.	
2	GET the following objects: maxSpecialFunctionOutputs.	Pass/Fail
3	VERIFY maxSpecialFunctionOutputs IS_NOT_LESS_THAN UserMinSpecialFunctionOutputs.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.38 Monitor Special Function Status

Test Procedure:	**Monitor Special Function Status**	
Description:	**This test case verifies that the ASC allows the management station to identify whether special functions are on or off.**	
Requirement(s):	• **3.5.2.2.8.2 Monitor Special Function Status**	
Variable(s):	**MaxRows** **maxSpecialFunctionOutputs** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSpecialFunctionOutputs.	Pass/Fail
1.1	RECORD this information as Table_Row.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: specialFunctionOutputStatus.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.11.39 Monitor Special Function Control Source

Test Procedure:	**Monitor Special Function Control Source**	
Description:	**This test case verifies that the ASC allows the management station to identify whether special functions are on or off and why any active special functions are on.**	
Requirement(s):	• **3.5.2.2.8.2 Monitor Special Function Status**	
Variable(s):	**MaxRows** **maxSpecialFunctionOutputs** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**

1	GET the following objects: maxSpecialFunctionOutputs.	Pass/Fail
1.1	RECORD this information as Table_Row.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: specialFunctionOutputControlSource.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12 Control Signal Operations

C.3.12.1 Enable/Disable Manual Backup

Test Procedure:	**Enable/Disable Manual Backup**
Description:	**This test case verifies that the ASC allows the management station to manually activate BackUp mode regardless of the BackUp timer and clear system control parameters.**
Requirement(s):	• **3.5.2.3.1.1 Enable/Disable Manual Backup**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDTION 'Manual Backup Mode is disabled.'	
2	ASSIGN unitManualBackup EQUALS 1.	
3	SET the following objects: unitManualBackup.	Pass/Fail
4	GET the following objects: unitManualBackup.	Pass/Fail
5	VERIFY unitManualBackup IS_EQUAL_TO 1.	Pass/Fail
6	DELAY 3 seconds	
7	ASSIGN unitManualBackup EQUALS 0.	
8	SET the following objects: unitManualBackup.	Pass/Fail
9	GET the following objects: unitManualBackup.	Pass/Fail
10	VERIFY unitControlV4 IS_EQUAL_TO 0.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.2 Control Global Minimum Recall

	Control Global Minimum Recall

Test Procedure:	
Description:	**This test case verifies that the ASC allows the management station to place calls for minimum recall on all phases.**
Requirement(s):	• **3.5.2.3.1.2 Control Global Minimum Recall**
Variable(s):	**OriginalUnitControl unitControlV4** **TestUnitControl unitControlV4**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitControlV4.	Pass/Fail
1.1	RECORD this information as OriginalUnitControl.	
2	ASSIGN TestUnitControl EQUALS OriginalUnitControl.	
3	ASSIGN TestUnitControl[0] EQUALS TestUnitControl XOR 128.	
4	ASSIGN unitControlV4 EQUALS TestUnitControl.	
5	SET the following objects: unitControlV4.	Pass/Fail
6	GET the following objects: unitControlV4.	Pass/Fail
7	VERIFY unitControlV4 IS_EQUAL_TO TestUnitControl.	Pass/Fail
8	ASSIGN unitControlV4 EQUALS OriginalUnitControl.	
9	SET the following objects: unitControlV4.	Pass/Fail
10	GET the following objects: unitControlV4.	Pass/Fail
11	VERIFY unitControlV4 IS_EQUAL_TO OriginalUnitControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.3 Control to Non-Actuated 1

Test Procedure:	**Control Call to Non-Actuated 1**
Description:	**This test case verifies that the ASC allows the management station to artificially activate the call to non-actuated 1 input.**
Requirement(s):	• **3.5.2.3.1.3 Control Call to Non-Actuated 1**
Variable(s):	**OriginalUnitControl unitControlV4** **TestUnitControl unitControlV4**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.

Test Step Number	Test Procedure	Results
1	GET the following objects: unitControlV4.	Pass/Fail
1.1	RECORD this information as OriginalUnitControl.	
2	ASSIGN TestUnitControl EQUALS OriginalUnitControl.	
3	ASSIGN TestUnitControl[0] EQUALS TestUnitControl XOR 64.	
4	ASSIGN unitControlV4 EQUALS TestUnitControl.	
5	SET the following objects: unitControlV4.	Pass/Fail
6	GET the following objects: unitControlV4.	Pass/Fail
7	VERIFY unitControlV4 IS_EQUAL_TO TestUnitControl.	Pass/Fail
8	ASSIGN unitControlV4 EQUALS OriginalUnitControl.	
9	SET the following objects: unitControlV4.	Pass/Fail
10	GET the following objects: unitControlV4.	Pass/Fail
11	VERIFY unitControlV4 IS_EQUAL_TO OriginalUnitControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.4 Control to Non-Actuated 2

Test Procedure:	**Control Call to Non-Actuated 2**
Description:	**This test case verifies that the ASC allows the management station to artificially activate the call to non-actuated 2 input.**
Requirement(s):	• **3.5.2.3.1.4 Control Call to Non-Actuated 2**
Variable(s):	**OriginalUnitControl unitControlV4** **TestUnitControl unitControlV4**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitControlV4.	Pass/Fail
1.1	RECORD this information as OriginalUnitControl.	
2	ASSIGN TestUnitControl EQUALS OriginalUnitControl.	
3	ASSIGN TestUnitControl[0] EQUALS TestUnitControl XOR 32.	
4	ASSIGN unitControlV4 EQUALS TestUnitControl.	
5	SET the following objects: unitControlV4.	Pass/Fail
6	GET the following objects: unitControlV4.	Pass/Fail
7	VERIFY unitControlV4 IS_EQUAL_TO TestUnitControl.	Pass/Fail

8	ASSIGN unitControlV4 EQUALS OriginalUnitControl.	
9	SET the following objects: unitControlV4.	Pass/Fail
10	GET the following objects: unitControlV4.	Pass/Fail
11	VERIFY unitControlV4 IS_EQUAL_TO OriginalUnitControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.5 Control Walk Rest Modifier

Test Procedure:	**Control Walk Rest Modifier**
Description:	**This test case verifies that the ASC allows the management station to set phases configured for non-actuated mode to rest-in-walk.**
Requirement(s):	• **3.5.2.3.1.5 Control Walk Rest Modifier**
Variable(s):	**OriginalUnitControl unitControlV4** **TestUnitControl unitControlV4**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitControlV4.	Pass/Fail
1.1	RECORD this information as OriginalUnitControl.	
2	ASSIGN TestUnitControl EQUALS OriginalUnitControl.	
3	ASSIGN TestUnitControl[0] EQUALS TestUnitControl XOR 16.	
4	ASSIGN unitControlV4 EQUALS TestUnitControl.	
5	SET the following objects: unitControlV4.	Pass/Fail
6	GET the following objects: unitControlV4.	Pass/Fail
7	VERIFY unitControlV4 IS_EQUAL_TO TestUnitControl.	Pass/Fail
8	ASSIGN unitControlV4 EQUALS OriginalUnitControl.	
9	SET the following objects: unitControlV4.	Pass/Fail
10	GET the following objects: unitControlV4.	Pass/Fail
11	VERIFY unitControlV4 IS_EQUAL_TO OriginalUnitControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.6 Control Interconnect

	Control Interconnect

Test Procedure:	
Description:	**This test case verifies that the ASC allows the management station to store interconnect inputs operate at a higher priority than the timebase control**
Requirement(s):	• **3.5.2.3.1.6 Control Interconnect**
Variable(s):	**OriginalUnitControl unitControlV4** **TestUnitControl unitControlV4**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitControlV4.	Pass/Fail
1.1	RECORD this information as OriginalUnitControl.	
2	ASSIGN TestUnitControl EQUALS OriginalUnitControl.	
3	ASSIGN TestUnitControl[0] EQUALS TestUnitControl XOR 8.	
4	ASSIGN unitControlV4 EQUALS TestUnitControl.	
5	SET the following objects: unitControlV4.	Pass/Fail
6	GET the following objects: unitControlV4.	Pass/Fail
7	VERIFY unitControlV4 IS_EQUAL_TO TestUnitControl.	Pass/Fail
8	ASSIGN unitControlV4 EQUALS OriginalUnitControl.	
9	SET the following objects: unitControlV4.	Pass/Fail
10	GET the following objects: unitControlV4.	Pass/Fail
11	VERIFY unitControlV4 IS_EQUAL_TO OriginalUnitControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.7 Activate System Timing Pattern Remotely

Test Procedure:	**Activate System Timing Pattern Remotely**
Description:	**This test case verifies that the ASC allows a management station to activate a timing pattern remotely.**
Requirement(s):	• **3.5.2.3.2.1 Activate System Timing Pattern Remotely**
Variable(s):	**OriginalPatternControl systemPatternControl** **TestPatternControl systemPatternControl**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: systemPatternControl.	Pass/Fail
1.1	RECORD this information as OriginalPatternControl.	
2	ASSIGN TestPatternControl EQUALS RANDOM (0 TO 255).	
2.1	IF TestPatternControl IS_EQUAL_TO OriginalPatternControl.	
2.1.1	GOTO step 2.	
3	ASSIGN systemPatternControl EQUALS TestPatternControl.	
4	SET the following objects: systemPatternControl.	Pass/Fail
5	GET the following objects: systemPatternControl.	Pass/Fail
6	VERIFY systemPatternControl IS_EQUAL_TO TestPatternControl.	Pass/Fail
7	ASSIGN systemPatternControl EQUALS OriginalPatternControl.	
8	SET the following objects: systemPatternControl.	Pass/Fail
9	GET the following objects: systemPatternControl.	Pass/Fail
10	VERIFY systemPatternControl IS_EQUAL_TO OriginalPatternControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.8 Control System Reference Point

Test Procedure:	**Control System Reference Point**	
Description:	**This test case verifies that the ASC allows a management station to configure a reference point for the called system pattern.**	
Requirement(s):	• **3.5.2.3.2.2 Control System Reference Point**	
Variable(s):	**OriginalSyncControl systemSyncControl** **TestSyncControl systemSyncControl**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: systemSyncControl.	Pass/Fail
1.1	RECORD this information as OriginalSyncControl.	
2	ASSIGN TestSyncControl EQUALS RANDOM (0 TO 255).	
2.1	IF OriginalSyncControl IS_EQUAL_TO TestSyncControl.	
2.1.1	GOTO step 2.	
3	ASSIGN systemSyncControl EQUALS TestSyncControl.	

4	SET the following objects: systemSyncControl.	Pass/Fail
5	GET the following objects: systemSyncControl.	Pass/Fail
6	VERIFY systemSyncControl IS_EQUAL_TO TestSyncControl.	Pass/Fail
7	ASSIGN systemSyncControl EQUALS OriginalSyncControl.	
8	SET the following objects: systemSyncControl.	Pass/Fail
9	GET the following objects: systemSyncControl.	Pass/Fail
10	VERIFY systemSyncControl IS_EQUAL_TO OriginalSyncControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.9 Control Phase Omits

Test Procedure:	**Control Phase Omits**	
Description:	**This test case verifies that the ASC allows a management station to omit phases from being serviced.**	
Requirement(s):	• **3.5.2.3.3.1 Control Phase Omits**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPhaseOmit** **TestPhaseOmit**	**maxPhaseGroups Int** **phaseControlGroupPhaseOmit** **phaseControlGroupPhaseOmit**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseControlGroupPhaseOmit.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPhaseOmit.	
4	ASSIGN TestPhaseOmit EQUALS RANDOM (0 TO 255).	
4.1	IF TestPhaseOmit IS_EQUAL_TO OriginalPhaseOmit.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseControlGroupPhaseOmit.Table_Row EQUALS TestPhaseOmit.	
6	SET the following objects: phaseControlGroupPhaseOmit.Table_Row.	Pass/Fail
7	GET the following objects: phaseControlGroupPhaseOmit.Table_Row.	Pass/Fail
8	VERIFY phaseControlGroupPhaseOmit.Table_Row IS_EQUAL_TO TestPhaseOmit.	Pass/Fail

9	ASSIGN phaseControlGroupPhaseOmit.Table_Row EQUALS OriginalPhaseOmit.	
10	SET the following objects: phaseControlGroupPhaseOmit.Table_Row.	Pass/Fail
11	GET the following objects: phaseControlGroupPhaseOmit.Table_Row.	Pass/Fail
12	VERIFY phaseControlGroupPhaseOmit.Table_Row IS_EQUAL_TO OriginalPhaseOmit.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.10 Control Pedestrian Phase Omits

Test Procedure:	**Control Pedestrian Phase Omits**	
Description:	**This test case verifies that the ASC allows a management station to omit pedestrian movements from being served.**	
Requirement(s):	• **3.5.2.3.3.2 Control Pedestrian Phase Omits**	
Variable(s):	**MaxRows** **Table_Row** **OriginalPedOmit** **TestPedOmit**	**maxPhaseGroups Int** **phaseControlGroupPhaseOmit** **phaseControlGroupPedOmit**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO 255).	
3	GET the following objects: phaseControlGroupPedOmit.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPedOmit.	
4	ASSIGN TestPedOmit EQUALS RANDOM (0 TO 255).	
4.1	IF TestPedOmit IS_EQUAL_TO OriginalPedOmit.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseControlGroupPedOmit.Table_Row EQUALS TestPedOmit.	
6	SET the following objects: phaseControlGroupPedOmit.Table_Row.	Pass/Fail
7	GET the following objects: phaseControlGroupPedOmit.Table_Row.	Pass/Fail
8	VERIFY phaseControlGroupPedOmit.Table_Row IS_EQUAL_TO TestPedOmit.	Pass/Fail
9	ASSIGN phaseControlGroupPedOmit.Table_Row EQUALS OriginalPedOmit.	
10	SET the following objects: phaseControlGroupPedOmit.Table_Row.	Pass/Fail
11	GET the following objects: phaseControlGroupPedOmit.Table_Row.	Pass/Fail
12	VERIFY phaseControlGroupPedOmit.Table_Row IS_EQUAL_TO OriginalPedOmit.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.11 Control Phase Holds

Test Procedure:	**Control Phase Hold**	
Description:	**This test case verifies that the ASC allows a management station to place holds on phases.**	
Requirement(s):	• **3.5.2.3.3.3 Control Phase Holds**	
Variable(s):	**MaxRows** **Table_Row** **OriginalHold** **TestHold**	**maxPhaseGroups** **Int** **phaseControlGroupHold** **phaseControlGroupHold**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseControlGroupHold.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalHold.	
4	ASSIGN TestHold EQUALS RANDOM (0 TO 255).	
4.1	IF TestHold IS_EQUAL_TO OriginalHold.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseControlGroupHold.Table_Row EQUALS TestHold.	
6	SET the following objects: phaseControlGroupHold.Table_Row.	Pass/Fail
7	GET the following objects: phaseControlGroupHold.Table_Row.	Pass/Fail
8	VERIFY phaseControlGroupHold.Table_Row IS_EQUAL_TO TestHold.	Pass/Fail
9	ASSIGN phaseControlGroupHold.Table_Row EQUALS OriginalHold.	
10	SET the following objects: phaseControlGroupHold.Table_Row.	Pass/Fail
11	GET the following objects: phaseControlGroupHold.Table_Row.	Pass/Fail
12	VERIFY phaseControlGroupHold.Table_Row IS_EQUAL_TO OriginalHold.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.12 Control Phase Force Offs

Test Procedure:	**Control Phase Force Offs**

Description:	**This test case verifies that the ASC allows a management station to force off phases.**	
Requirement(s):	• **3.5.2.3.3.4 Control Phase Force Offs**	
Variable(s):	**MaxRows** **Table_Row** **OriginalForceOff** **TestForceOff**	**maxPhaseGroups** **Int** **phaseControlGroupForceOff** **phaseControlGroupForceOff**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as OriginalForceOff.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseControlGroupForceOff.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalForceOff.	
4	ASSIGN TestForceOff EQUALS RANDOM (0 TO 255).	
4.1	IF TestForceOff IS_EQUAL_TO OriginalForceOff.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseControlGroupForceOff.Table_Row EQUALS TestForceOff.	
6	SET the following objects: phaseControlGroupForceOff.Table_Row.	Pass/Fail
7	GET the following objects: phaseControlGroupForceOff.Table_Row.	Pass/Fail
8	VERIFY phaseControlGroupForceOff.Table_Row IS_EQUAL_TO TestForceOff.	Pass/Fail
9	ASSIGN phaseControlGroupForceOff.Table_Row EQUALS OriginalForceOff.	
10	SET the following objects: phaseControlGroupForceOff.Table_Row.	Pass/Fail
11	GET the following objects: phaseControlGroupForceOff.Table_Row.	Pass/Fail
12	VERIFY phaseControlGroupForceOff.Table_Row IS_EQUAL_TO OriginalForceOff.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.13 Control Phase Vehicle Calls

Test Procedure:	**Control Phase Vehicle Calls**
Description:	**This test case verifies that the ASC allows a management station to place calls for service on phases remotely.**
Requirement(s):	• **3.5.2.3.3.5 Control Phase Vehicle Calls**

Variable(s):	**MaxRows** **maxPhaseGroups** **Table_Row** **Int** **OriginalCall** **phaseControlGroupVehCall** **TestCall** **phaseControlGroupVehCall**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseControlGroupVehCall.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalCall.	
4	ASSIGN TestCall EQUALS RANDOM (0 TO 255).	
4.1	IF TestCall IS_EQUAL_TO OriginalCall.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseControlGroupVehCall.Table_Row EQUALS TestCall.	
6	SET the following objects: phaseControlGroupVehCall.Table_Row.	Pass/Fail
7	GET the following objects: phaseControlGroupVehCall.Table_Row.	Pass/Fail
8	VERIFY phaseControlGroupVehCall.Table_Row IS_EQUAL_TO TestCall.	Pass/Fail
9	ASSIGN phaseControlGroupVehCall.Table_Row EQUALS OriginalCall.	
10	SET the following objects: phaseControlGroupVehCall.Table_Row.	Pass/Fail
11	GET the following objects: phaseControlGroupVehCall.Table_Row.	Pass/Fail
12	VERIFY phaseControlGroupVehCall.Table_Row IS_EQUAL_TO OriginalCall.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.14 Control Phase Pedestrian Calls

Test Procedure:	**Control Phase Pedestrian Calls**
Description:	**This test case verifies that the ASC allows a management station to place calls for pedestrian service on phases remotely.**
Requirement(s):	• **3.5.2.3.3.6 Control Phase Pedestrian Calls**
Variable(s):	**MaxRows** **maxPhaseGroups** **Table_Row** **Int** **OriginalPedCall** **phaseControlGroupPedCall** **TestPedCall** **phaseControlGroupPedCall**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPhaseGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: phaseControlGroupPedCall.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPedCall.	
4	ASSIGN TestPedCall EQUALS RANDOM (0 TO 255).	
4.1	IF TestPedCall IS_EQUAL_TO OriginalPedCall.	
4.1.1	GOTO step 4.	
5	ASSIGN phaseControlGroupPedCall.Table_Row EQUALS TestPedCall.	
6	SET the following objects: phaseControlGroupPedCall.Table_Row.	Pass/Fail
7	GET the following objects: phaseControlGroupPedCall.Table_Row.	Pass/Fail
8	VERIFY phaseControlGroupPedCall.Table_Row IS_EQUAL_TO TestPedCall.	Pass/Fail
9	ASSIGN phaseControlGroupPedCall.Table_Row EQUALS OriginalPedCall.	
10	SET the following objects: phaseControlGroupPedCall.Table_Row.	Pass/Fail
11	GET the following objects: phaseControlGroupPedCall.Table_Row.	Pass/Fail
12	VERIFY phaseControlGroupPedCall.Table_Row IS_EQUAL_TO OriginalPedCall.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.15 Activate Preempt Remotely

Test Procedure:	Activate Preempt Remotely	
Description:	**This test case verifies that the ASC allows a management station activate preempts remotely.**	
Requirement(s):	• **3.5.2.3.4 Activate Preempt Remotely**	
Variable(s):	**MaxRows** **Table_Row** **OriginalState** **TestState**	**maxPreempts** **Int** **preemptControlState** **preemptControlState**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPreempts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: preemptControlState.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalState.	
4	IF OriginalState IS_EQUAL_TO 0.	
4.1	ASSIGN TestState EQUALS 1.	
4.2	GOTO step 6.	
5	ASSIGN TestState EQUALS 0.	
6	ASSIGN preemptControlState.Table_Row EQUALS TestState.	
7	SET the following objects: preemptControlState.Table_Row.	Pass/Fail
8	GET the following objects: preemptControlState.Table_Row.	Pass/Fail
9	VERIFY preemptControlState.Table_Row IS_EQUAL_TO TestState.	Pass/Fail
10	ASSIGN preemptControlState.Table_Row EQUALS OriginalState.	
11	SET the following objects: preemptControlState.Table_Row.	Pass/Fail
12	GET the following objects: preemptControlState.Table_Row.	Pass/Fail
13	VERIFY preemptControlState.Table_Row IS_EQUAL_TO OriginalState.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.16 Control Ring Stop Time

Test Procedure:	**Control Ring Stop Time**
Description:	**This test case verifies that the ASC allows a management station to stop timing by ring.**
Requirement(s):	• **3.5.2.3.5.1 Control Ring Stop Time**
Variable(s):	**MaxRows** **Table_Row** **OriginalStopTime** **TestStopTime** **maxRingControlGroups** **Int** **ringControlGroupStopTime** **ringControlGroupStopTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupStopTime.Table_Row.	Pass/Fail

3.1	RECORD this information as OriginalStopTime.	
4	ASSIGN TestStopTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestStopTime IS_EQUAL_TO OriginalStopTime.	
4.1.1	GOTO step 4.	
5	ASSIGN ringControlGroupStopTime.Table_Row EQUALS TestStopTime.	
6	SET the following objects: ringControlGroupStopTime.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupStopTime.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupStopTime.Table_Row IS_EQUAL_TO TestStopTime.	Pass/Fail
9	ASSIGN ringControlGroupStopTime.Table_Row EQUALS OriginalStopTime.	
10	SET the following objects: ringControlGroupStopTime.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupStopTime.Table_Row.	Pass/Fail
12	VERIFY ringControlGroupStopTime.Table_Row IS_EQUAL_TO OriginalStopTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.17 Control Ring Force Offs

Test Procedure:	**Control Ring Force Offs**	
Description:	**This test case verifies that the ASC allows a management station to force off phases remotely using ring settings.**	
Requirement(s):	• **3.5.2.3.5.2 Control Ring Force Offs**	
Variable(s):	**MaxRows** **Table_Row** **OriginalForceOff** **TestForceOff**	**maxRingControlGroups** **Int** **ringControlGroupForceOff** **ringControlGroupForceOff**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupForceOff.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalForceOff.	
4	ASSIGN TestForceOff EQUALS RANDOM (0 TO 255).	
4.1	IF TestForceOff IS_EQUAL_TO OriginalForceOff.	
4.1.1	GOTO step 4.	

5	ASSIGN ringControlGroupForceOff.Table_Row EQUALS TestForceOff.	
6	SET the following objects: ringControlGroupForceOff.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupForceOff.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupForceOff.Table_Row IS_EQUAL_TO TestForceOff.	Pass/Fail
9	ASSIGN ringControlGroupForceOff.Table_Row EQUALS OriginalForceOff.	
10	SET the following objects: ringControlGroupForceOff.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupForceOff.Table_Row.	Pass/Fail
12	VERIFY ringControlGroupForceOff.Table_Row IS_EQUAL_TO OriginalForceOff.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.18 Control Ring Maximum 2 Time Settings

Test Procedure:	**Control Ring Maximum 2 Time Settings**	
Description:	**This test case verifies that the ASC allows a management station to activate the phase maximum2 settings using rings.**	
Requirement(s):	• **3.5.2.3.5.3 Control Ring Maximum 2 Time Settings**	
Variable(s):	**MaxRows** **OriginalGroupMax2** **TestGroupMax2** **Table_Row**	**maxRingControlGroups** **ringControlGroupMax2** **ringControlGroupMax2** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupMax2.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalGroupMax2.	
4	ASSIGN TestGroupMax2 EQUALS RANDOM (0 TO 255).	
4.1	IF TestGroupMax2 IS_EQUAL_TO OriginalGroupMax2.	
4.1.1	GOTO step 4.	
5	ASSIGN ringControlGroupMax2.Table_Row EQUALS TestGroupMax2.	
6	SET the following objects: ringControlGroupMax2.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupMax2.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupMax2.Table_Row IS_EQUAL_TO TestGroupMax2.	Pass/Fail

9	ASSIGN ringControlGroupMax2.Table_Row EQUALS OriginalGroupMax2.	
10	SET the following objects: ringControlGroupMax2.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupMax2.Table_Row.	Pass/Fail
12	VERIFY ringControlGroupMax2.Table_Row IS_EQUAL_TO OriginalGroupMax2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.19 Control Ring Maximum 3 Time Settings

Test Procedure:	**Control Ring Maximum 3 Time Settings**	
Description:	**This test case verifies that the ASC allows a management station to activate the phase maximum3 settings using rings.**	
Requirement(s):	• **3.5.2.3.5.4 Control Ring Maximum 3 Time Settings**	
Variable(s):	**MaxRows** **OriginalGroupMax3** **TestGroupMax3** **Table_Row**	**maxRingControlGroups** **ringControlGroupMax3** **ringControlGroupMax3** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupMax3.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalGroupMax3.	
4	ASSIGN TestGroupMax3 EQUALS RANDOM (0 TO 255).	
4.1	IF TestGroupMax3 IS_EQUAL_TO OriginalGroupMax3.	
4.1.1	GOTO step 4.	
5	ASSIGN ringControlGroupMax3.Table_Row EQUALS TestGroupMax3.	
6	SET the following objects: ringControlGroupMax3.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupMax3.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupMax3.Table_Row IS_EQUAL_TO TestGroupMax3.	Pass/Fail
9	ASSIGN ringControlGroupMax3.Table_Row EQUALS OriginalGroupMax3.	
10	SET the following objects: ringControlGroupMax3.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupMax3.Table_Row.	Pass/Fail

12	VERIFY ringControlGroupMax3.Table_Row IS_EQUAL_TO OriginalGroupMax3.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.20 Control Ring Maximum Inhibit Settings

Test Procedure:	**Control Ring Maximum Inhibit Settings**
Description:	**This test case verifies that the ASC allows a management station to activate the phase maxInhibit settings using rings.**
Requirement(s):	• **3.5.2.3.5.5 Control Ring Maximum Inhibit Settings**
Variable(s):	**MaxRows** **maxRingControlGroups** **OriginalGroupMaxInhibit** **ringControlGroupMaxInhibit** **TestGroupControlInhibit** **ringControlGroupMaxInhibit Int** **Table_Row**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupMaxInhibit.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalGroupMaxInhibit.	
4	ASSIGN TestGroupControlInhibit EQUALS RANDOM (0 TO 255).	
4.1	IF TestGroupControlInhibit IS_EQUAL_TO OriginalGroupMaxInhibit.	
4.1.1	GOTO step 4.	
5	ASSIGN ringControlGroupMaxInhibit.Table_Row EQUALS TestGroupControlInhibit.	
6	SET the following objects: ringControlGroupMaxInhibit.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupMaxInhibit.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupMaxInhibit.Table_Row IS_EQUAL_TO TestGroupControlInhibit.	Pass/Fail
9	ASSIGN ringControlGroupMaxInhibit.Table_Row EQUALS OriginalGroupMaxInhibit.	
10	SET the following objects: ringControlGroupMaxInhibit.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupMaxInhibit.Table_Row.	Pass/Fail
12	VERIFY ringControlGroupMaxInhibit.Table_Row IS_EQUAL_TO OriginalGroupMaxInhibit.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.12.21 Control Ring Pedestrian Recycle Settings

Test Procedure:	**Control Ring Pedestrian Recycle Settings**
Description:	**This test case verifies that the ASC allows a management station to enable pedestrian recycle on phases using ring settings.**
Requirement(s):	• **3.5.2.3.5.6 Control Ring Pedestrian Recycle Settings**
Variable(s):	**MaxRows** maxRingControlGroups **Table_Row** Int **OriginalPedRecycle** ringControlGroupPedRecycle **TestPedRecycle** ringControlGroupPedRecycle
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupPedRecycle.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPedRecycle.	
4	ASSIGN TestPedRecycle EQUALS RANDOM (0 TO 255).	
4.1	IF TestPedRecycle IS_EQUAL_TO OriginalPedRecycle.	
4.1.1	GOTO step 4.	
5	ASSIGN ringControlGroupPedRecycle.Table_Row EQUALS TestPedRecycle.	
6	SET the following objects: ringControlGroupPedRecycle.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupPedRecycle.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupPedRecycle.Table_Row IS_EQUAL_TO TestPedRecycle.	Pass/Fail
9	ASSIGN ringControlGroupPedRecycle.Table_Row EQUALS OriginalPedRecycle.	
10	SET the following objects: ringControlGroupPedRecycle.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupPedRecycle.Table_Row.	Pass/Fail
12	VERIFY ringControlGroupPedRecycle.Table_Row IS_EQUAL_TO OriginalPedRecycle.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.22 Control Ring Red Rest Settings

Test Procedure:	**Control Ring Red Rest Settings**	
Description:	**This test case verifies that the ASC allows a management station to configure phases to rest in red using ring settings.**	
Requirement(s):	• **3.5.2.3.5.7 Control Ring Red Rest Settings**	
Variable(s):	**MaxRows** **Table_Row** **OriginalRedRest** **TestRedRest**	**maxRingControlGroups** **Int** **ringControlGroupRedRest** **ringControlGroupRedRest**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupRedRest.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalRedRest.	
4	ASSIGN TestRedRest EQUALS RANDOM (0 TO 255).	
4.1	IF TestRedRest IS_EQUAL_TO OriginalRedRest.	
4.1.1	GOTO step 4.	
5	ASSIGN ringControlGroupRedRest.Table_Row EQUALS TestRedRest.	
6	SET the following objects: ringControlGroupRedRest.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupRedRest.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupRedRest.Table_Row IS_EQUAL_TO TestRedRest.	Pass/Fail
9	ASSIGN ringControlGroupRedRest.Table_Row EQUALS OriginalRedRest.	
10	SET the following objects: ringControlGroupRedRest.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupRedRest.Table_Row.	Pass/Fail
12	VERIFY ringControlGroupRedRest.Table_Row IS_EQUAL_TO OriginalRedRest.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.23 Control Ring Pedestrian Recycle Settings

Test Procedure:	**8.12 Control Ring Red Clearance Omit Settings**

Description:	**This test case verifies that the ASC allows a management station to omit red clearance intervals.**
Requirement(s):	• **3.5.2.3.5.8 Control Ring Red Clearance Omit Settings**
Variable(s):	**MaxRows** **maxRingControlGroups** **OriginalOmitRedClear** **ringControlGroupOmitRedClear** **TestOmitRedClear** **ringControlGroupOmitRedClear** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRingControlGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: ringControlGroupOmitRedClear.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOmitRedClear.	
4	ASSIGN TestOmitRedClear EQUALS RANDOM (0 TO 255).	
4.1	IF TestOmitRedClear IS_EQUAL_TO OriginalOmitRedClear.	
4.1.1	GOTO step 4.	
5	ASSIGN ringControlGroupOmitRedClear.Table_Row EQUALS RANDOM (0 TO 255).	
6	SET the following objects: ringControlGroupOmitRedClear.Table_Row.	Pass/Fail
7	GET the following objects: ringControlGroupOmitRedClear.Table_Row.	Pass/Fail
8	VERIFY ringControlGroupOmitRedClear.Table_Row IS_EQUAL_TO TestOmitRedClear.	Pass/Fail
9	ASSIGN ringControlGroupOmitRedClear.Table_Row EQUALS OriginalOmitRedClear.	
10	SET the following objects: ringControlGroupOmitRedClear.Table_Row.	Pass/Fail
11	GET the following objects: ringControlGroupOmitRedClear.Table_Row.	Pass/Fail
12	VERIFY ringControlGroupOmitRedClear.Table_Row IS_EQUAL_TO OriginalOmitRedClear.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.24 Activate Special Function Remotely

Test Procedure:	**Activate Special Function Remotely**

Description:	**This test case verifies that the ASC allows a management station to enable or disable special function outputs.**
Requirement(s):	• **3.5.2.3.6 Activate Special Function Remotely**
Variable(s):	**MaxRows** **maxSpecialFunctionOutputs** **Table_Row** **Int** **OriginalOutputControl** **specialFunctionOutputControl** **TestOutputControl** **specialFunctionOutputControl**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSpecialFunctionOutputs.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOutputControl.	
4	IF OriginalOutputControl IS_EQUAL_TO 0.	
4.1	ASSIGN TestOutputControl EQUALS 1.	
4.2	GOTO step 6.	
5	ASSIGN TestOutputControl EQUALS 0.	
6	ASSIGN specialFunctionOutputControl.Table_Row EQUALS TestOutputControl.	
7	SET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
8	GET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
9	VERIFY specialFunctionOutputControl.Table_Row IS_EQUAL_TO TestOutputControl.	Pass/Fail
10	ASSIGN specialFunctionOutputControl.Table_Row EQUALS OriginalOutputControl.	
11	SET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
12	GET the following objects: specialFunctionOutputControl.Table_Row.	Pass/Fail
13	VERIFY specialFunctionOutputControl.Table_Row IS_EQUAL_TO OriginalOutputControl.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.25 Remote Manual Control

Test Procedure:	**Remote Manual Control**
Description:	**This test case verifies that the ASC allows a management station to command the signal controller to advance to the next interval.**

Requirement(s):	• **3.5.2.3.7.1 Enable Manual Control** • **3.5.2.3.7.2 Remote Manual Control Advance** • **3.5.2.3.7.3 Configure Manual Control Timeout**
Variable(s):	**TestTimeOut** **unitMCETimeout**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: unitMCETimeout.	Pass/Fail
2	ASSIGN TestTimeOut EQUALS RANDOM (10 TO 65535)	
3	ASSIGN unitMCETimeout EQUALS TestTimeOut	
4	SET the following objects: unitMCETimeout.	Pass/Fail
5	DELAY 3 Seconds	
6	GET the following objects: unitMCETimeout.	Pass/Fail
7	VERIFY unitMCETimeout IS_NOT_LESS_THAN (TestTimeOut – 3)	Pass/Fail
8	GET the following objects: unitMCEIntAdv.	Pass/Fail
9	ASSIGN unitMCEIntAdv EQUALS 1.	
10	SET the following objects: unitMCEIntAdv.	Pass/Fail
11	DELAY 1 Second	
12	GET the following objects: unitMCEIntAdv.	Pass/Fail
13	VERIFY unitMCEIntAdv IS_EQUAL_TO 0.	Pass/Fail
14	ASSIGN unitMCETimeout EQUALS 0.	
15	SET the following objects: unitMCETimeOut.	Pass/Fail
16	GET the following objects: unitMCETimeout.	Pass/Fail
17	VERIFY unitMCETimeout IS_EQUAL_TO 0.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.12.26 Enable/Disable Automatic Pedestrian Clearance Setting

Test Procedure:	**Enable/Disable Automatic Pedestrian Clearance Setting**
Description:	**This test case verifies that the ASC allows a management station to protect the Pedestrian Clearance Interval from being terminated by an Internal Advance input when Manual Control is active.**
Requirement(s):	• **3.5.2.3.7.4 Enable/Disable Automatic Pedestrian Clearance Setting**
Variable(s):	**OriginalClear** **unitAutoPedestrianClear** **TestClear** **unitAutoPedestrianClear**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results

1	GET the following objects: unitAutoPedestrianClear.	Pass/Fail
1.1	RECORD this information as OriginalClear.	
2	IF OriginalClear IS_EQUAL_TO 1.	
2.1	ASSIGN TestClear EQUALS 2.	
3	IF OriginalClear IS_EQUAL_TO 2.	
3.1	ASSIGN TestClear EQUALS 1.	
4	ASSIGN unitAutoPedestrianClear EQUALS TestClear.	
5	SET the following objects: unitAutoPedestrianClear.	Pass/Fail
6	GET the following objects: unitAutoPedestrianClear.	Pass/Fail
7	VERIFY unitAutoPedestrianClear IS_EQUAL_TO TestClear.	Pass/Fail
8	ASSIGN unitAutoPedestrianClear EQUALS OriginalClear.	
9	SET the following objects: unitAutoPedestrianClear.	Pass/Fail
10	GET the following objects: unitAutoPedestrianClear.	Pass/Fail
11	VERIFY unitAutoPedestrianClear IS_EQUAL_TO OriginalClear.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13 Manage Detector Configuration

C.3.13.1 Configure Vehicle Travel Mode

Test Procedure:	**3.05 Configure Vehicle Travel Mode**	
Description:	**This test case verifies that the ASC allows a management station to configure the travel mode for a detector.**	
Requirement(s):	• **3.5.3.1.1.1 Configure Vehicle Travel Mode**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTravelMode** **TestTravelMode**	**maxVehicleDetectors** **Int** **vehicleDetectorTravelMode** **vehicleDetectorTravelMode**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorTravelMode.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTravelMode.	
4	ASSIGN TestTravelMode EQUALS RANDOM (2 TO 4).	
4.1	IF TestTravelMode IS_EQUAL_TO OriginalTravelMode.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorTravelMode.Table_Row EQUALS TestTravelMode.	
6	SET the following objects: vehicleDetectorTravelMode.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorTravelMode.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorTravelMode.Table_Row IS_EQUAL_TO TestTravelMode.	Pass/Fail
9	ASSIGN vehicleDetectorTravelMode.Table_Row EQUALS OriginalTravelMode.	
10	SET the following objects: vehicleDetectorTravelMode.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorTravelMode.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorTravelMode.Table_Row IS_EQUAL_TO OriginalTravelMode.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.2 Configure Vehicle Detector Description

Test Procedure:	**Configure Vehicle Detector Description**		
Description:	**This test case verifies that the ASC allows a management station to configure descriptions for vehicle detectors.**		
Requirement(s):	• **3.5.3.1.1.2 Configure Vehicle Detector Description**		
Variable(s):	**MaxRows** **Table_Row** **OriginalDescription**	**maxVehicleDetectors** **Int** **vehicleDetectorDescription**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: maxVehicleDetectors.		Pass/Fail
1.1	RECORD this information as MaxRows.		
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).		
3	GET the following objects: vehicleDetectorDescription.Table_Row.		Pass/Fail
3.1	RECORD this information as OriginalDescription		
4	ASSIGN vehicleDetectorDescription.Table_Row EQUALS 'Test'.		
5	SET the following objects: vehicleDetectorDescription.Table_Row.		Pass/Fail
6	GET the following objects: vehicleDetectorDescription.Table_Row.		Pass/Fail
7	VERIFY vehicleDetectorDescription.Table_Row IS_EQUAL_TO 'Test'.		Pass/Fail
8	ASSIGN vehicleDetectorDescription.Table_Row EQUALS OrignalDescription.		
9	SET the following objects: vehicleDetectorDescription.Table_Row.		Pass/Fail
10	GET the following objects: vehicleDetectorDescription.Table_Row.		Pass/Fail
11	VERIFY vehicleDetectorDescription.Table_Row IS_EQUAL_TO OriginalDescription.		Pass/Fail
Test Procedure Results			
Tested By:		**Date Tested:**	**Pass/Fail**
Test Procedure Notes:			

C.3.13.3 Configure Vehicle Detector Yellow Lock Call Enabled

Test Procedure:	**Configure Vehicle Detector Yellow Lock Call Enabled**	
Description:	**This test case verifies that the ASC allows a management station to set a vehicle detector to set a vehicle detector to lock a call to the assigned phase if an actuation occurs during the Yellow Change Interval**	
Requirement(s):	• **3.5.3.1.1.3 Configure Vehicle Detector Yellow Lock Call Enabled**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions**

	TestOptions	**vehicleDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 4.	
5	ASSIGN vehicleDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN vehicleDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.4 Configure Vehicle Detector Red Lock Call Enabled

Test Procedure:	**Configure Vehicle Detector Red Lock Call Enabled**	
Description:	**This test case verifies that the ASC allows a management station to set a vehicle detector to lock a call to the assigned phase if an actuation occurs while the phase is not timing Green or Yellow Change intervals.**	
Requirement(s):	• **3.5.3.1.1.4 Configure Vehicle Detector Red Lock Call Enabled**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions** **vehicleDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results

1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 8.	
5	ASSIGN vehicleDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN vehicleDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.5 Configure Vehicle Detector Passage Enabled

Test Procedure:	**Configure Vehicle Detector Passage Enabled**	
Description:	**This test case verifies that the ASC allows a management station to set the associated phase passage timer to reset for the duration of a vehicle detector actuation if the phase is in the Green interval.**	
Requirement(s):	• **3.5.3.1.1.5 Configure Vehicle Detector Passage Enabled**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions** **vehicleDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	

4	ASSIGN TestOptions EQUALS OriginalOptions XOR 16.	
5	ASSIGN vehicleDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN vehicleDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.6 Configure Vehicle Detector Added Initial Time Enabled

Test Procedure:	**Configure Vehicle Detector Added Initial Time Enabled**	
Description:	**This test case verifies that the ASC allows a management station to set the ASC to use vehicle detector actuation counts for added initial calculations.**	
Requirement(s):	• **3.5.3.1.1.6 Configure Vehicle Detector Added Initial Time Enabled**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions** **vehicleDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 32.	
5	ASSIGN vehicleDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN vehicleDetectorOptions.Table_Row EQUALS OriginalOptions.	

10	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.7 Configure Vehicle Detector Queue Enabled

Test Procedure:	**Configure Vehicle Detector Queue Enabled**	
Description:	**This test case verifies that the ASC allows a management station to set a vehicle detector to detect the presence of queues.**	
Requirement(s):	• **3.5.3.1.1.7 Configure Vehicle Detector Queue Enabled**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions** **vehicleDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 64.	
5	ASSIGN vehicleDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN vehicleDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.13.8 Configure Vehicle Detector Call Enabled

Test Procedure:	**Configure Vehicle Detector Call Enabled**	
Description:	**This test case verifies that the ASC allows a management station to store if a call is placed for vehicle service upon actuation of a vehicle detector while the phase is not timing the Green interval.**	
Requirement(s):	• **3.5.3.1.1.8 Configure Vehicle Detector Call Enabled**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions** **vehicleDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 128.	
5	ASSIGN vehicleDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN vehicleDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.9 Configure Vehicle Detector Call Phase

Test Procedure:	**Configure Vehicle Detector Call Phase**	
Description:	**This test case verifies that the ASC allows a management station to assign call phases to vehicle detectors.**	
Requirement(s):	• **3.5.3.1.1.9 Configure Vehicle Detector Call Phase**	
Variable(s):	**MaxRows** **Table_Row** **OriginalCallPhase** **TestCallPhase** **MaxPhases**	**maxVehicleDetectors** **Int** **vehicleDetectorCallPhase** **vehicleDetectorCallPhase** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of phases the ASC supports'.	
1.1	RECORD this information as MaxPhases.	
2	GET the following objects: maxVehicleDetectors.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: vehicleDetectorCallPhase.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalCallPhase.	
5	ASSIGN TestCallPhase EQUALS RANDOM (1 TO MaxPhases).	
5.1	IF TestCallPhase IS_EQUAL_TO OriginalCallPhase.	
5.1.1	GOTO step 5.	
6	ASSIGN vehicleDetectorCallPhase.Table_Row EQUALS TestCallPhase.	
7	SET the following objects: vehicleDetectorCallPhase.Table_Row.	Pass/Fail
8	GET the following objects: vehicleDetectorCallPhase.Table_Row.	Pass/Fail
9	VERIFY vehicleDetectorCallPhase.Table_Row IS_EQUAL_TO TestCallPhase.	Pass/Fail
10	ASSIGN vehicleDetectorCallPhase.Table_Row EQUALS OriginalCallPhase.	
11	SET the following objects: vehicleDetectorCallPhase.Table_Row.	Pass/Fail
12	GET the following objects: vehicleDetectorCallPhase.Table_Row.	Pass/Fail
13	VERIFY vehicleDetectorCallPhase.Table_Row IS_EQUAL_TO OriginalCallPhase.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.10 Configure Vehicle Detector Switch Phase

Test Procedure:	**Configure Vehicle Detector Switch Phase**
Description:	**This test case verifies that the ASC allows a management station to assign a switch phase to a vehicle detector to be switched upon turning green to when its assigned phase is YELLOW or RED.**
Requirement(s):	• **3.5.3.1.1.10 Configure Vehicle Detector Switch Phase**
Variable(s):	**MaxRows** **maxVehicleDetectors** **OriginalSwitchPhase** **vehicleDetectorSwitchPhase** **TestSwitchPhase** **vehicleDetectorSwitchPhase** **Table_Row** **Int** **MaxPhases** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of phases the ASC supports'.	
1.1	RECORD this information as MaxPhases.	
2	GET the following objects: maxVehicleDetectors.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: vehicleDetectorSwitchPhase.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalSwitchPhase.	
5	ASSIGN TestSwitchPhase EQUALS RANDOM (1 TO MaxPhases).	
5.1	IF TestSwitchPhase IS_EQUAL_TO OriginalSwitchPhase.	
5.1.1	GOTO step 4.	
6	ASSIGN vehicleDetectorSwitchPhase.Table_Row EQUALS TestSwitchPhase.	
7	SET the following objects: vehicleDetectorSwitchPhase.Table_Row.	Pass/Fail
8	GET the following objects: vehicleDetectorSwitchPhase.Table_Row.	Pass/Fail
9	VERIFY vehicleDetectorSwitchPhase.Table_Row IS_EQUAL_TO TestSwitchPhase.	Pass/Fail
10	ASSIGN vehicleDetectorSwitchPhase.Table_Row EQUALS OriginalSwitchPhase.	
11	SET the following objects: vehicleDetectorSwitchPhase.Table_Row.	Pass/Fail
12	GET the following objects: vehicleDetectorSwitchPhase.Table_Row.	Pass/Fail
13	VERIFY vehicleDetectorSwitchPhase.Table_Row IS_EQUAL_TO OriginalSwitchPhase.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

Test Procedure Notes:

C.3.13.11 Configure Vehicle Detector Delay Time

Test Procedure:	**Configure Vehicle Detector Delay Time**	
Description:	**This test case verifies that the ASC allows a management station the period a detector actuation shall be delayed when the phase is not GREEN.**	
Requirement(s):	• **3.5.3.1.1.11 Configure Vehicle Detector Delay Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDetectorDelay** **TestDetectorDelay**	**maxVehicleDetectors** **Int** **vehicleDetectorDelay** **vehicleDetectorDelay**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorDelay.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDetectorDelay.	
4	ASSIGN TestDetectorDelay EQUALS RANDOM (0 TO 2550).	
4.1	IF TestDetectorDelay IS_EQUAL_TO OriginalDetectorDelay.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorDelay.Table_Row EQUALS TestDetectorDelay.	
6	SET the following objects: vehicleDetectorDelay.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorDelay.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorDelay.Table_Row IS_EQUAL_TO TestDetectorDelay.	Pass/Fail
9	ASSIGN vehicleDetectorDelay.Table_Row EQUALS OriginalDetectorDelay.	
10	SET the following objects: vehicleDetectorDelay.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorDelay.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorDelay.Table_Row IS_EQUAL_TO OriginalDetectorDelay.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.12 Configure Vehicle Detector Extend Time

Test Procedure:	**Configure Vehicle Detector Extend Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the time that actuation for a vehicle detector is extended from the point of termination**	
Requirement(s):	• **3.5.3.1.1.12 Configure Vehicle Detector Extend Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDetectorExtend** **TestDetectorExtend**	**maxVehicleDetectors** **Int** **vehicleDetectorExtend** **vehicleDetectorExtend**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorExtend.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDetectorExtend.	
4	ASSIGN TestDetectorExtend EQUALS RANDOM (0 TO 255).	
4.1	IF TestDetectorExtend IS_NOT_EQUAL_TO OriginalDetectorExtend.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorExtend.Table_Row EQUALS TestDetectorExtend.	
6	SET the following objects: vehicleDetectorExtend.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorExtend.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorExtend.Table_Row IS_EQUAL_TO TestDetectorExtend.	Pass/Fail
9	ASSIGN vehicleDetectorExtend.Table_Row EQUALS OriginalDetectorExtend.	
10	SET the following objects: vehicleDetectorExtend.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorExtend.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorExtend.Table_Row IS_EQUAL_TO OriginalDetectorExtend.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.13 Configure Vehicle Detector Queue Limit Time

Test Procedure:	**Configure Vehicle Detector Queue Limit Time**
Description:	**This test case verifies that the ASC allows a management station to configure the length of time that an actuation from a queue detector may continue into the phase green.**
Requirement(s):	• **3.5.3.1.1.13 Configure Vehicle Detector Queue Limit Time**
Variable(s):	**MaxRows** — **maxVehicleDetectors** **Table_Row** — **Int** **OriginalQueueLimit** — **vehicleDetectorQueueLimit** **TestQueueLimit** — **vehicleDetectorQueueLimit**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorQueueLimit.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalQueueLimit.	
4	ASSIGN TestQueueLimit EQUALS RANDOM (0 TO 255).	
4.1	IF vehicleDetectorQueueLimit.Table_Row IS_EQUAL_TO OriginalQueueLimit.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorQueueLimit.Table_Row EQUALS TestQueueLimit.	
7	SET the following objects: vehicleDetectorQueueLimit.Table_Row.	Pass/Fail
8	GET the following objects: vehicleDetectorQueueLimit.Table_Row.	Pass/Fail
9	VERIFY vehicleDetectorQueueLimit.Table_Row IS_EQUAL_TO TestQueueLimit.	Pass/Fail
10	ASSIGN vehicleDetectorQueueLimit.Table_Row EQUALS OriginalQueueLimit.	
11	SET the following objects: vehicleDetectorQueueLimit.Table_Row.	Pass/Fail
12	GET the following objects: vehicleDetectorQueueLimit.Table_Row.	Pass/Fail
13	VERIFY vehicleDetectorQueueLimit.Table_Row IS_EQUAL_TO OriginalQueueLimit.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.14 Configure Vehicle Detector No Activity Time

Test Procedure:	**Configure Vehicle Detector No Activity Time**
Description:	**This test case verifies that the ASC allows a management station to configure the time in minutes that must elapse before a vehicle detector that does not exhibit an actuation is considered Failed.**
Requirement(s):	• **3.5.3.1.1.14 Configure Vehicle Detector No Activity Time**
Variable(s):	**MaxRows** maxVehicleDetectors **Table_Row** Int **OriginalNoActivity** vehicleDetectorNoActivity **TestNoActivity** vehicleDetectorNoActivity
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorNoActivity.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalNoActivity.	
4	ASSIGN TestNoActivity EQUALS RANDOM (0 TO 65535).	
4.1	IF TestNoActivity IS_NOT_EQUAL_TO OriginalNoActivity.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorNoActivity.Table_Row EQUALS TestNoActivity.	
6	SET the following objects: vehicleDetectorNoActivity.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorNoActivity.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorNoActivity.Table_Row IS_EQUAL_TO TestNoActivity.	Pass/Fail
9	ASSIGN vehicleDetectorNoActivity.Table_Row EQUALS OriginalNoActivity.	
10	SET the following objects: vehicleDetectorNoActivity.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorNoActivity.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorNoActivity.Table_Row IS_EQUAL_TO OriginalNoActivity.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.15 Configure Vehicle Detector Maximum Presence Time

Test Procedure:	**Configure Vehicle Detector Maximum Presence Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the maximum time in minutes that a detector actuation may exhibited continuously before the vehicle detector is considered Failed.**	
Requirement(s):	• **3.5.3.1.1.15 Configure Vehicle Detector Maximum Presence Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalMaxPresence** **TestMaxPresence**	**maxVehicleDetectors** **Int** **vehicleDetectorMaxPresence** **vehicleDetectorMaxPresence**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorMaxPresence.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalMaxPresence.	
4	ASSIGN TestMaxPresence EQUALS RANDOM (0 TO 255).	
4.1	IF TestMaxPresence IS_EQUAL_TO OriginalMaxPresence.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorMaxPresence.Table_Row EQUALS TestMaxPresence.	
6	SET the following objects: vehicleDetectorMaxPresence.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorMaxPresence.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorMaxPresence.Table_Row IS_EQUAL_TO TestMaxPresence.	Pass/Fail
9	ASSIGN vehicleDetectorMaxPresence.Table_Row EQUALS OriginalMaxPresence.	
10	SET the following objects: vehicleDetectorMaxPresence.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorMaxPresence.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorMaxPresence.Table_Row IS_EQUAL_TO OriginalMaxPresence.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.16 Configure Vehicle Detector Erratic Counts

Test Procedure:	**Configure Vehicle Detector Erratic Counts**
Description:	**This test case verifies that the ASC allows a management station to configure the maximum number of actuations a vehicle detector may exhibit within a minute before being considered Failed.**
Requirement(s):	• **3.5.3.1.1.16 Configure Vehicle Detector Erratic Counts**
Variable(s):	**MaxRows** **maxVehicleDetectors** **Table_Row** **Int** **OriginalErraticCounts** **vehicleDetectorErraticCounts** **TestErraticCounts** **vehicleDetectorErraticCounts**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorErraticCounts.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalErraticCounts.	
4	ASSIGN TestErraticCounts EQUALS RANDOM (0 TO 255).	
4.1	IF TestErraticCounts IS_NOT_EQUAL_TO OriginalErraticCounts.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorErraticCounts.Table_Row EQUALS TestErraticCounts.	
6	SET the following objects: vehicleDetectorErraticCounts.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorErraticCounts.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorErraticCounts.Table_Row IS_EQUAL_TO TestErraticCounts.	Pass/Fail
9	ASSIGN vehicleDetectorErraticCounts.Table_Row EQUALS OriginalErraticCounts.	
10	SET the following objects: vehicleDetectorErraticCounts.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorErraticCounts.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorErraticCounts.Table_Row IS_EQUAL_TO OriginalErraticCounts.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.17 Configure Vehicle Detector Fail Time

Test Procedure:	**Configure Vehicle Detector Fail Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the vehicle detector fail time.**	
Requirement(s):	• **3.5.3.1.1.17 Configure Vehicle Detector Fail Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalFailTime** **TestFailTime**	**maxVehicleDetectors** **Int** **vehicleDetectorFailTime** **vehicleDetectorFailTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorFailTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalFailTime.	
4	ASSIGN TestFailTime EQUALS RANDOM (0 TO 255).	
4.1	IF TestFailTime IS_EQUAL_TO OriginalFailTime.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorFailTime.Table_Row EQUALS TestFailTime.	
6	SET the following objects: vehicleDetectorFailTime.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorFailTime.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorFailTime.Table_Row IS_EQUAL_TO TestFailTime.	Pass/Fail
9	ASSIGN vehicleDetectorFailTime.Table_Row EQUALS OriginalFailTime.	
10	SET the following objects: vehicleDetectorFailTime.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorFailTime.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorFailTime.Table_Row IS_EQUAL_TO OriginalFailTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.18 Configure Pedestrian Detector Description

Test Procedure:	**Configure Pedestrian Detector Description**

Description:	**This test case verifies that the ASC allows a management station to configure descriptions for pedestrian detectors.**	
Requirement(s):	• **3.5.3.1.3.1 Configure Pedestrian Detector Description**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDescription**	**maxPedestrianDetectors Int** **pedestrianDetectorDescription**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorDescription.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDescription	
4	ASSIGN pedestrianDetectorDescription.Table_Row EQUALS 'Test'.	
5	SET the following objects: pedestrianDetectorDescription.Table_Row.	Pass/Fail
6	GET the following objects: pedestrianDetectorDescription.Table_Row.	Pass/Fail
7	VERIFY pedestrianDetectorDescription.Table_Row IS_EQUAL_TO 'Test'.	Pass/Fail
8	ASSIGN pedestrianDetectorDescription.Table_Row EQUALS OrignalDescription.	
9	SET the following objects: pedestrianDetectorDescription.Table_Row.	Pass/Fail
10	GET the following objects: pedestrianDetectorDescription.Table_Row.	Pass/Fail
11	VERIFY pedestrianDetectorDescription.Table_Row IS_EQUAL_TO OriginalDescription.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.19 Configure Pedestrian Detector Call Phase

Test Procedure:	**Configure Pedestrian Detector Call Phase**	
Description:	**This test case verifies that the ASC allows a management station to assign call phases for pedestrian detector inputs.**	
Requirement(s):	• **3.5.2.1.13.1.2 Configure APS Push Button to Phase Association** • **3.5.3.1.3.2 Configure Pedestrian Detector Call Phase**	
Variable(s):	**MaxRows** **Table_Row** **OriginalCallPhase** **MaxPhases** **TestCallPhase**	**maxPedestrianDetectors Int** **pedestrianDetectorCallPhase Int** **pedestrianDetectorCallPhase**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of phases supported by the ASC'.	
1.1	RECORD this information as MaxPhases.	
2	GET the following objects: maxPedestrianDetectors.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: pedestrianDetectorCallPhase.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalCallPhase.	
5	ASSIGN TestCallPhase EQUALS RANDOM (0 TO MaxPhases).	
5.1	IF TestCallPhase IS_EQUAL_TO OriginalCallPhase.	
5.1.1	GOTO step 5.	
6	ASSIGN pedestrianDetectorCallPhase.Table_Row EQUALS TestCallPhase.	
7	SET the following objects: pedestrianDetectorCallPhase.Table_Row.	Pass/Fail
8	GET the following objects: pedestrianDetectorCallPhase.Table_Row.	Pass/Fail
9	VERIFY pedestrianDetectorCallPhase.Table_Row IS_EQUAL_TO TestCallPhase.	Pass/Fail
10	ASSIGN pedestrianDetectorCallPhase.Table_Row EQUALS OriginalCallPhase.	
11	SET the following objects: pedestrianDetectorCallPhase.Table_Row.	Pass/Fail
12	GET the following objects: pedestrianDetectorCallPhase.Table_Row.	Pass/Fail
13	VERIFY pedestrianDetectorCallPhase.Table_Row IS_EQUAL_TO OriginalCallPhase.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.20 Configure Pedestrian Detector No Activity Time

Test Procedure:	**Configure Pedestrian Detector No Activity Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the time in minutes that must elapse before a pedestrian detector that does not exhibit an actuation is considered Failed.**	
Requirement(s):	• **3.5.3.1.3.3 Configure Pedestrian Detector No Activity Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalNoActivity**	**maxPedestrianDetectors** **Int** **pedestrianDetectorNoActivity**

	TestNoActivity	**pedestrianDetectorNoActivity**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorNoActivity.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalNoActivity.	
4	ASSIGN Table_Row EQUALS RANDOM (0 TO 65535).	
4.1	IF TestNoActivity IS_EQUAL_TO OriginalNoActivity.	
4.1.1	GOTO step 4.	
5	ASSIGN pedestrianDetectorNoActivity.Table_Row EQUALS TestNoActivity.	
6	SET the following objects: pedestrianDetectorNoActivity.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorNoActivity.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorNoActivity.Table_Row IS_EQUAL_TO TestNoActivity.	Pass/Fail
9	ASSIGN pedestrianDetectorNoActivity.Table_Row EQUALS OriginalNoActivity.	
10	SET the following objects: pedestrianDetectorNoActivity.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorNoActivity.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorNoActivity.Table_Row IS_EQUAL_TO OriginalNoActivity.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.21 Configure Pedestrian Detector Maximum Presence Fault Time

Test Procedure:	**Configure Pedestrian Detector Maximum Presence Time**	
Description:	**This test case verifies that the ASC allows a management station to configure the maximum time in minutes a detector actuation may exhibited continuously before the pedestrian detector is considered Failed.**	
Requirement(s):	• **3.5.3.1.3.4 Configure Pedestrian Detector Maximum Presence Time**	
Variable(s):	**MaxRows** **Table_Row** **OriginalMaxPresence**	**maxPedestrianDetectors** **Int** **pedestrianDetectorMaxPresence**

	TestMaxPresence	**pedestrianDetectorMaxPresence**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorMaxPresence.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalMaxPresence.	
4	ASSIGN TestMaxPresence EQUALS RANDOM (0 TO 255).	
4.1	IF TestMaxPresence IS_EQUAL_TO OriginalMaxPresence.	
4.1.1	GOTO step 4.	
5	ASSIGN maxPedestrianDetectors.Table_Row EQUALS TestMaxPresence.	
6	SET the following objects: pedestrianDetectorMaxPresence.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorMaxPresence.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorMaxPresence.Table_Row IS_EQUAL_TO TestMaxPresence.	Pass/Fail
9	ASSIGN maxPedestrianDetectors.Table_Row EQUALS OriginalMaxPresence.	
10	SET the following objects: pedestrianDetectorMaxPresence.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorMaxPresence.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorMaxPresence.Table_Row IS_EQUAL_TO OriginalMaxPresence.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.22 Configure Pedestrian Detector Erratic Counts

Test Procedure:	**Configure Pedestrian Detector Erratic Counts**	
Description:	**This test case verifies that the ASC allows a management station to configure the maximum number of actuations a pedestrian detector must exhibit within a minute before being considered Failed.**	
Requirement(s):	• **3.5.3.1.3.5 Configure Pedestrian Detector Erratic Counts**	
Variable(s):	**MaxRows** **Table_Row** **OriginalErraticCounts**	**maxPedestrianDetectors** **Int** **pedestrianDetectorErraticCounts**

	TestErraticCounts	**pedestrianDetectorErraticCounts**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorErraticCounts.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalErraticCounts.	
4	ASSIGN TestErraticCounts EQUALS RANDOM (0 TO 255).	
4.1	IF TestErraticCounts IS_EQUAL_TO OriginalErraticCounts.	
4.1.1	GOTO step 4.	
5	ASSIGN pedestrianDetectorErraticCounts.Table_Row EQUALS TestErraticCounts.	
6	SET the following objects: pedestrianDetectorErraticCounts.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorErraticCounts.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorErraticCounts.Table_Row IS_EQUAL_TO TestErraticCounts.	Pass/Fail
9	ASSIGN pedestrianDetectorErraticCounts.Table_Row EQUALS OriginalErraticCounts.	
10	SET the following objects: pedestrianDetectorErraticCounts.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorErraticCounts.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorErraticCounts.Table_Row IS_EQUAL_TO OriginalErraticCounts.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.23 Configure Pedestrian Detector Non-Lock Calls

Test Procedure:	**Configure Pedestrian Detector Non-Lock Calls**	
Description:	**This test case verifies that the ASC allows a management station to configure a pedestrian detector for non-locked calls for pedestrian timings.**	
Requirement(s):	• **3.5.3.1.3.6 Configure Pedestrian Detector Non-Lock Calls**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **pedestrianDetectorOptions** **pedestrianDetectorOptions**

Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 4.	
5	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.24 Configure Pedestrian Detector for Alternate Pedestrian Timing

Test Procedure:	Configure Pedestrian Detector for Alternate Pedestrian Timing
Description:	This test case verifies that the ASC allows a management station to configure a pedestrian detector to place calls for alternate pedestrian timing instead of normal pedestrian timing.
Requirement(s):	• 3.5.2.1.13.1.4 Configure Pedestrian Detector for Alternate Pedestrian Timing
Variable(s):	MaxRows — maxVehicleDetectors Table_Row — Int OriginalOptions — pedestrianDetectorOptions TestOptions — pedestrianDetectorOptions
Pass/Fail Criteria:	The device under test shall pass every verification step in this test case to pass the test case.

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 2.	
5	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.25 Configure Pedestrian Detector for Presence Detection

Test Procedure:	**Configure Pedestrian Detector for Presence Detection**	
Description:	**This test case verifies that the ASC allows a management station to set a pedestrian detector to detect the presence of a pedestrian in a crosswalk instead of detecting a pedestrian call for service.**	
Requirement(s):	• **3.5.3.1.3.7 Configure Pedestrian Detector for Presence Detection**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors Int** **pedestrianDetectorOptions** **pedestrianDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail

1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestOptions EQUALS OriginalOptions XOR 1.	
5	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.26 Configure Pedestrian Detector for Delayed Walk

Test Procedure:	**Configure Pedestrian Detector for Delayed Walk**	
Description:	**This test case verifies that the ASC allows a management station to set a pedestrian detector to activate delayed walk settings for pedestrian phases.**	
Requirement(s):	• **3.5.3.1.3.8 Configure Pedestrian Detector for Delayed Walk**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **pedestrianDetectorOptions** **pedestrianDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	

4	ASSIGN TestChannelFlash EQUALS OriginalOptions AND 11.	
4.1	NOTE 'Toggle Bit 4 LOW because Bits 3 & 4 of pedestrianDetectorOptions cannot both be TRUE'.	
5	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.27 Configure Pedestrian Detector for Advanced Walk

Test Procedure:	**Configure Pedestrian Detector for Advanced Walk**	
Description:	**This test case verifies that the ASC allows a management station to set a pedestrian detector to activate advanced walk settings for pedestrian phases.**	
Requirement(s):	• **3.5.3.1.3.9 Configure Pedestrian Detector for Advanced Walk**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions** **TestOptions**	**maxVehicleDetectors** **Int** **pedestrianDetectorOptions** **pedestrianDetectorOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions.	
4	ASSIGN TestChannelFlash EQUALS OriginalOptions AND 7.	

4.1	NOTE 'Toggle Bit 3 LOW because Bits 3 & 4 of pedestrianDetectorOptions cannot both be TRUE'.	
5	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS TestOptions.	
6	SET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorOptions.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO TestOptions.	Pass/Fail
9	ASSIGN pedestrianDetectorOptions.Table_Row EQUALS OriginalOptions.	
10	SET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
11	GET the following objects: pedestrianDetectorOptions.Table_Row.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorOptions.Table_Row.Table_Row IS_EQUAL_TO OriginalOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.28 Determine Maximum Number of Vehicle Detectors

Test Procedure:	**Determine Maximum Number of Vehicle Detectors**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of vehicle detectors supported by ASC and verifies that the ASC supports the minimum number of vehicle detectors required by the user.**	
Requirement(s):	• **3.5.3.1.5.1 Determine Maximum Number of Vehicle Detectors**	
Variable(s):	**UserMinVehicleDetectors maxVehicleDetectors**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of vehicle detectors required as specified in FR ID 3.5.3.1.5.1 of the PRL.'	
1.1	RECORD this information as UserMinVehicleDetectors.	
2	GET the following objects: maxVehicleDetectors.	Pass/Fail
3	VERIFY maxVehicleDetectors IS_NOT_LESS_THAN UserMinVehicleDetectors.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.13.29 Determine Maximum Number of Vehicle Detector Sets

Test Procedure:	**Determine Maximum Number of Vehicle Detector Sets**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of vehicle detector sets supported by ASC and verifies that the ASC supports the minimum number of vehicle detector sets required by the user.**
Requirement(s):	• **3.5.3.1.5.2 Determine Maximum Number of Vehicle Detector Sets**
Variable(s):	**UserMinVehicleDetectorSets maxVehicleDetectorSets**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of vehicle detector sets required as specified in FR ID 3.5.3.1.5.2 of the PRL.'	
1.1	RECORD this information as UserMinVehicleDetectorSets.	
2	GET the following objects: maxVehicleDetectorSets.	Pass/Fail
3	VERIFY maxVehicleDetectorSets IS_NOT_LESS_THAN UserMinVehicleDetectorSets.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.30 Determine Maximum Number of Pedestrian Detectors

Test Procedure:	**Determine Maximum Number of Pedestrian Detectors**
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of pedestrian detectors supported by ASC and verifies that the ASC supports the minimum number of pedestrian detectors required by the user.**
Requirement(s):	• **3.5.3.1.5.3 Determine Maximum Number of Pedestrian Detectors**
Variable(s):	**UserMinPedestrianDetectors maxPedestrianDetectors**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of pedestrian detectors required as specified in FR ID 3.5.3.1.5.3 of the PRL.'	
1.1	RECORD this information as UserMinPedestrianDetectors.	
2	GET the following objects: maxPedestrianDetectors.	Pass/Fail
3	VERIFY maxPedestrianDetectors IS_NOT_LESS_THAN UserMinPedestrianDetectors.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.31 Determine Maximum Number of Pedestrian Detector Sets

Test Procedure:	**Determine Maximum Number of Pedestrian Detector Sets**	
Description:	**This test case verifies that the ASC allows a management station to determine the maximum number of pedestrian detector sets supported by ASC and verifies that the ASC supports the minimum number of pedestrian detector sets required by the user.**	
Requirement(s):	• **3.5.3.1.5.4 Determine Maximum Number of Pedestrian Detector Sets**	
Variable(s):	**UserMinPedestrianDetectorSets** **maxPedestrianDetectorSets**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	USER-ACTION 'Determine the number of pedestrian detector sets required as specified in FR ID 3.5.3.1.5.4 of the PRL.'	
1.1	RECORD this information as UserMinPedestrianDetectorSets.	
2	GET the following objects: maxPedestrianDetectorSets.	Pass/Fail
3	VERIFY maxPedestrianDetectorSets IS_NOT_LESS_THAN UserMinPedestrianDetectorSets.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.32 Monitor Active Vehicle Detector Actuations

Test Procedure:	**Monitor Active Vehicle Detector Actuations**	
Description:	**This test case verifies that the ASC allows a management station to identify vehicle detectors that are actuated.**	
Requirement(s):	• **3.5.3.2.1 Monitor Active Vehicle Detector Actuations**	
Variable(s):	**MaxRows** **maxVehicleDetectorGroups** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxVehicleDetectorGroups.	Pass/Fail
1.1	RECORD this information as Table_Row.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: vehicleDetectorStatusGroupActive.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.33 Monitor Active Pedestrian Detector Actuations

Test Procedure:	**Monitor Active Pedestrian Detector Actuations**	
Description:	**This test case verifies that the ASC allows a management station to identify pedestrian detectors that are actuated.**	
Requirement(s):	• **3.5.3.2.2 Monitor Active Pedestrian Detector Actuations**	
Variable(s):	**MaxRows** **maxPedestrianDetectorGroups** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**

1	GET the following objects: maxPedestrianDetectorGroups.	Pass/Fail
1.1	RECORD this information as Table_Row.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: pedestrianDetectorStatusGroupActive.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.34 Monitor Vehicle Detector Alarm Status

Test Procedure:	**Monitor Vehicle Detector Alarm Status**		
Description:	**This test case verifies that the ASC allows a management station to identify vehicle detectors with active alarms.**		
Requirement(s):	• **3.5.3.3.1.1 Monitor Vehicle Detector Alarm Status**		
Variable(s):	**MaxRows** **Table_Row**	**maxVehicleDetectorGroups** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectorGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: vehicleDetectorStatusGroupAlarms.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.35 Monitor Vehicle Detector Faults from Controller

Test Procedure:	**Monitor Vehicle Detector Faults from Controller**

Description:	**This test case verifies that the ASC allows a management station to identify vehicle detector faults as identified by the ASC.**
Requirement(s):	• **3.5.3.3.1.2 Monitor Vehicle Detector Faults from Controller**
Variable(s):	**MaxRows** **maxVehicleDetectors** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: vehicleDetectorAlarms.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.36 Monitor Vehicle Detector Faults from Detector

Test Procedure:	**Monitor Vehicle Detector Faults from Detector**
Description:	**This test case verifies that the ASC allows a management station to identify vehicle detector faults as identified by the detector.**
Requirement(s):	• **3.5.3.3.1.3 Monitor Vehicle Detector Faults from Detector**
Variable(s):	**MaxRows** **maxVehicleDetectors** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: vehicleReportedDetectorAlarms.Table_Row.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.13.37 Monitor Pedestrian Detector Alarm Status

Test Procedure:	**Monitor Pedestrian Detector Alarm Status**	
Description:	**This test case verifies that the ASC allows a management station to identify pedestrian detectors with active alarms.**	
Requirement(s):	• **3.5.3.3.2.1 Monitor Pedestrian Detector Alarm Status**	
Variable(s):	**MaxRows** **Table_Row**	**maxPedestrianDetectorGroups** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectorGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1.1	GET the following objects: pedestrianDetectorStatusGroupAlarms.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.38 Monitor Pedestrian Detector Faults

Test Procedure:	**Monitor Pedestrian Detector Faults**	
Description:	**This test case verifies that the ASC allows a management station to identify pedestrian detector faults as identified by the ASC.**	
Requirement(s):	• **3.5.3.3.2.2 Monitor Pedestrian Detector Faults from Controller**	
Variable(s):	**MaxRows** **Table_Row**	**maxVehicleDetectors** **Int**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: vehicleDetectorAlarms.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.39 Control Vehicle Detector Reset

Test Procedure:	**Control Vehicle Detector Reset**	
Description:	**This test case verifies that the ASC allows a management station to reset a vehicle detector.**	
Requirement(s):	• **3.5.3.4.1 Control Vehicle Detector Reset**	
Variable(s):	**MaxRows** **maxVehicleDetectors** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	ASSIGN vehicleDetectorReset.Table_Row EQUALS 1.	
4	SET the following objects: vehicleDetectorReset.Table_Row.	Pass/Fail
5	USER-ACTION 'Verify the vehicle detector has reset.'	
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.40 Control Pedestrian Detector Reset

	Control Pedestrian Detector Reset

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to reset a pedestrian detector.**
Requirement(s):	• **3.5.3.4.2 Control Pedestrian Detector Reset**
Variable(s):	**MaxRows** **maxPedestrianDetectors** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	ASSIGN pedestrianDetectorReset.Table_Row EQUALS 1.	
4	SET the following objects: pedestrianDetectorReset.Table_Row.	Pass/Fail
5	USER-ACTION 'Verify the pedestrian detector has reset.'	
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.13.41 Control Detector Diagnostic Reset

Test Procedure:	**Control Detector Diagnostic Reset**
Description:	**This test case verifies that the ASC allows a management station to reset detector diagnostic counters**
Requirement(s):	• **3.5.3.4.3 Control Detector Diagnostic Reset**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	ASSIGN unitDetectorDiagnosticReset_Row EQUALS 1.	
2	SET the following objects: unitDetectorDiagnosticReset.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.13.42 Control Vehicle Detector Actuation

Test Procedure:	**Control Vehicle Detector Actuation**
Description:	**This test case verifies that the ASC allows a management station to place or remove artificial actuations on vehicle detectors.**
Requirement(s):	• **3.5.3.4.4 Control Vehicle Detector Actuation**
Variable(s):	**MaxRows** maxVehicleDetectorGroups **Table_Row** Int **OriginalActuation** vehicleDetectorControlGroupActuation **TestActuation** vehicleDetectorControlGroupActuation
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectorGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorControlGroupActuation.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalActuation.	
4	ASSIGN TestActuation EQUALS RANDOM (0 TO 255).	
4.1	IF TestActuation IS_EQUAL_TO OriginalActuation.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorControlGroupActuation.Table_Row EQUALS TestActuation.	
6	SET the following objects: vehicleDetectorControlGroupActuation.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorControlGroupActuation.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorControlGroupActuation.Table_Row IS_EQUAL_TO TestActuation.	Pass/Fail
9	ASSIGN vehicleDetectorControlGroupActuation.Table_Row EQUALS OriginalActuation.	
10	SET the following objects: vehicleDetectorControlGroupActuation.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorControlGroupActuation.Table_Row.	Pass/Fail

12	VERIFY vehicleDetectorControlGroupActuation.Table_Row IS_EQUAL_TO OriginalActuation.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

C.3.13.43 Control Pedestrian Detector Actuation

Test Procedure:	**Control Pedestrian Detector Actuation**
Description:	**This test case verifies that the ASC allows a management station to place or remove artificial actuations on pedestrian detectors.**
Requirement(s):	• **3.5.3.4.5 Control Pedestrian Detector Actuation**
Variable(s):	**MaxRows** — **maxPedestrianDetectorGroups** **Table_Row** — **Int** **OriginalActuation** — **pedestrianDetectorControlGroup Actuation** **TestActuation** — **pedestrianDetectorControlGroup Actuation**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectorDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: pedestrianDetectorControlGroupActuation.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalActuation.	
4	ASSIGN TestActuation EQUALS RANDOM (0 TO 255).	
4.1	IF TestActuation IS_EQUAL_TO OriginalActuation.	
4.1.1	GOTO step 4.	
5	ASSIGN pedestrianDetectorControlGroupActuation.Table_Row EQUALS TestActuation.	
6	SET the following objects: pedestrianDetectorControlGroupActuation.Table_Row.	Pass/Fail
7	GET the following objects: pedestrianDetectorControlGroupActuation.Table_Row.	Pass/Fail
8	VERIFY pedestrianDetectorControlGroupActuation.Table_Row IS_EQUAL_TO TestActuation.	Pass/Fail
9	ASSIGN pedestrianDetectorControlGroupActuation.Table_Row EQUALS OriginalActuation.	
10	SET the following objects: pedestrianDetectorControlGroupActuation.Table_Row.	Pass/Fail

11	GET the following objects: pedestrianDetectorControlGroupActuation.Table_Row.	Pass/Fail
12	VERIFY pedestrianDetectorControlGroupActuation.Table_Row IS_EQUAL_TO OriginalActuation.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14 Detector Data Collection

C.3.14.1 Monitor Detector Data Sequence

Test Procedure:	**Monitor Detector Data Sequence**	
Description:	**This test case verifies that the ASC allows a management station determine the sequence number for volume/occupancy data collection that is used to detect duplicate or missing reports.**	
Requirement(s):	• **3.5.3.5.1.1 Monitor Detector Data Sequence**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: volumeOccupancySequence.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.2 Monitor Vehicle Volume Data

Test Procedure:	**Monitor Vehicle Volume Data**	
Description:	**This test case verifies that the ASC allows a management station to retrieve volume data recorded by the ASC.**	
Requirement(s):	• **3.5.3.5.1.2 Monitor Vehicle Volume Data**	
Variable(s):	**MaxRows** **Table_Row**	**maxVehicleDetectors** **Int**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: detectorVolume.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.3 Monitor Vehicle Occupancy Data

Test Procedure:	**Monitor Vehicle Occupancy Data**
Description:	**This test case verifies that the ASC allows a management station to retrieve occupancy data recorded by the ASC.**
Requirement(s):	• **3.5.3.5.1.3 Monitor Vehicle Occupancy Data**
Variable(s):	**MaxRows** **maxVehicleDetectors** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: detectorOccupancy.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.4 Monitor Vehicle Average Speed

Test Procedure:	**Monitor Vehicle Average Speed**

Description:	**This test case verifies that the ASC allows a management station to retrieve average speed data recorded by the ASC.**		
Requirement(s):	• **3.5.3.5.1.3 Monitor Vehicle Average Speed**		
Variable(s):	**MaxRows** **Table_Row**	**maxVehicleDetectors** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: maxVehicleDetectors.		Pass/Fail
1.1	RECORD this information as MaxRows.		
2	FOR Table_Row from 1 to MaxRows.		
2.1	GET the following objects: detectorAvgSpeed.Table_Row.		Pass/Fail
Test Procedure Results			
Tested By:		**Date Tested:**	**Pass/Fail**
Test Procedure Notes:			

C.3.14.5 Monitor Vehicle Detector Sample Time

Test Procedure:	**Monitor Vehicle Detector Data Sample Time**		
Description:	**This test case verifies that the ASC allows a management station to determine the time the sample period of the detector ends.**		
Requirement(s):	• **3.5.3.5.1.5 Monitor Vehicle Detector Data Sample Time**		
Variable(s):			
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: detectorSampleTime.		Pass/Fail
Test Procedure Results			
Tested By:		**Date Tested:**	**Pass/Fail**
Test Procedure Notes:			

C.3.14.6 Monitor Vehicle Detector Sample Duration

	Monitor Vehicle Detector Data Sample Duration

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to determine the duration of the data collection period for the vehicle detectors.**
Requirement(s):	• **3.5.3.5.1.6 Monitor Vehicle Detector Data Sample Duration**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: detectorSampleDuration.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.7 Monitor Vehicle Detector Sample Duration

Test Procedure:	**Monitor Pedestrian Detector Data Sequence**
Description:	**This test case verifies that the ASC allows a management station to identify the current sequence number for pedestrian detector data collection.**
Requirement(s):	• **3.5.3.5.2.1 Monitor Pedestrian Detector Data Sequence**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: pedestrianDetectorSequence.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.8 Monitor Pedestrian Counts

Test Procedure:	**Monitor Pedestrian Counts**	
Description:	**This test case verifies that the ASC allows a management station to view the number of pedestrians currently detected within a detection zone during a defined sample period.**	
Requirement(s):	• **3.5.3.5.2.2 Monitor Pedestrian Counts**	
Variable(s):	**MaxRows** **maxPedestrianDetectors** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: pedestrianDetectorVolume.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.9 Monitor Pedestrian Detector Actuations

Test Procedure:	**Monitor Pedestrian Detector Actuations**	
Description:	**This test case verifies that the ASC allows a management station to identify the number of pedestrian actuations collected during a sample period.**	
Requirement(s):	• **3.5.3.5.2.3 Monitor Pedestrian Detector Actuations**	
Variable(s):	**MaxRows** **maxPedestrianDetectors** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	

2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: pedestrianDetectorActuations.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.10 Monitor Pedestrian Services

Test Procedure:	**Monitor Pedestrian Services**
Description:	**This test case verifies that the ASC allows a management station to identify the number of pedestrian services (number of times the ped transitioned from don't walk to walk) during the defined sample period.**
Requirement(s):	• **3.5.3.5.2.3 Monitor Pedestrian Services**
Variable(s):	**MaxRows** **maxPedestrianDetectors** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxPedestrianDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: pedestrianDetectorServices.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.11 Monitor Pedestrian Detector Data Sample Time

Test Procedure:	**Monitor Pedestrian Detector Data Sample Time**
Description:	**This test case verifies that the ASC allows a management station to identify the time of the pedestrian detector data collection period ends.**
Requirement(s):	• **3.5.3.5.2.5 Monitor Pedestrian Detector Data Sample Time**

Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: pedestrianDetectorSampleTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.12 Monitor Pedestrian Detector Data Sample Duration

Test Procedure:	**Monitor Pedestrian Detector Data Sample Duration**	
Description:	**This test case verifies that the ASC allows a management station to identify the duration of the pedestrian data collection period.**	
Requirement(s):	• **3.5.3.5.2.6 Monitor Pedestrian Detector Data Sample Duration**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: pedestrianDetectorSampleDuration.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.13 Configure Vehicle Detector Data Sample Period

Test Procedure:	**Configure Vehicle Detector Data Sample Period**
Description:	**This test case verifies that that ASC allows a management station to configure the volume/occupancy/speed collection period.**
Requirement(s):	• **3.5.3.5.3.1 Configure Detector Data Sample Period**

Variable(s):	**OriginalPeriod volumeOccupancyPeriod TestPeriod volumeOccupancyPeriod**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: volumeOccupancyPeriod.	Pass/Fail
1.1	RECORD this information as OriginalPeriod.	
2	ASSIGN TestPeriod EQUALS RANDOM (0 TO 3600).	
2.1	IF TestPeriod IS_EQUAL_TO OriginalPeriod.	
2.1.1	GOTO step 2.	
3	ASSIGN volumeOccupancyPeriod EQUALS TestPeriod.	
4	SET the following objects: volumeOccupancyPeriod.	Pass/Fail
5	GET the following objects: volumeOccupancyPeriod.	Pass/Fail
6	VERIFY volumeOccupancyPeriod IS_EQUAL_TO TestPeriod.	Pass/Fail
7	ASSIGN volumeOccupancyPeriod EQUALS OriginalPeriod.	
8	SET the following objects: volumeOccupancyPeriod.	Pass/Fail
9	GET the following objects: volumeOccupancyPeriod.	Pass/Fail
10	VERIFY volumeOccupancyPeriod IS_EQUAL_TO OriginalPeriod.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.14 Configure Pedestrian Detector Data Sample Period

Test Procedure:	**Configure Pedestrian Data Collection Sample Period**
Description:	**This test case verifies that that ASC allows a management station to configure the sample period for collecting pedestrian detector data.**
Requirement(s):	• **3.5.3.5.3.2 Configure Pedestrian Data Collection Sample Period**
Variable(s):	**OriginalPeriod pedestrianDetectorPeriod TestPeriod pedestrianDetectorPeriod**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: pedestrianDetectorPeriod.	Pass/Fail
1.1	RECORD this information as OriginalPeriod.	
2	ASSIGN TestPeriod EQUALS RANDOM (0 TO 3600).	

2.1	IF TestPeriod IS_EQUAL_TO OriginalPeriod.	
2.1.1	GOTO step 2.	
3	ASSIGN v pedestrianDetectorPeriod EQUALS TestPeriod.	
4	SET the following objects: pedestrianDetectorPeriod.	Pass/Fail
5	GET the following objects: pedestrianDetectorPeriod.	Pass/Fail
6	VERIFY pedestrianDetectorPeriod IS_EQUAL_TO TestPeriod.	Pass/Fail
7	ASSIGN pedestrianDetectorPeriod EQUALS OriginalPeriod.	
8	SET the following objects: pedestrianDetectorPeriod.	Pass/Fail
9	GET the following objects: pedestrianDetectorPeriod.	Pass/Fail
10	VERIFY pedestrianDetectorPeriod IS_EQUAL_TO OriginalPeriod.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.15 Configure Vehicle Speed Detectors

Test Procedure:	**Configure Vehicle Speed Detectors**	
Description:	**This test case verifies that the ASC allows a management station to enable/disable vehicle detectors collecting speed data.**	
Requirement(s):	• **3.5.3.5.3.3 Configure Vehicle Speed Detectors**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions2** **TestOptions2**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions2** **vehicleDetectorOptions2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions2.	
4	ASSIGN TestOptions2 EQUALS OriginalOptions2 XOR 1.	
5	ASSIGN vehicleDetectorOptions2.Table_Row EQUALS TestOptions2.	
6	SET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions2.Table_Row IS_EQUAL_TO TestOptions2.	Pass/Fail
9	ASSIGN vehicleDetectorOptions2.Table_Row EQUALS OriginalOptions2.	

10	SET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions2.Table_Row IS_EQUAL_TO OriginalOptions2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.16 Configure Single Detector Speed Mode

Test Procedure:	**Configure Single Detector Speed Mode**	
Description:	**This test case verifies that the ASC allows a management station to configure a detector to calculate speed without a paired detector.**	
Requirement(s):	• **3.5.3.5.3.4 Configure Single Detector Speed Mode**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions2** **TestOptions2**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions2** **vehicleDetectorOptions2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions2.	
4	ASSIGN TestOptions2 EQUALS OriginalOptions2 XOR 4.	
5	ASSIGN vehicleDetectorOptions2.Table_Row EQUALS TestOptions2.	
6	SET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions2.Table_Row IS_EQUAL_TO TestOptions2.	Pass/Fail
9	ASSIGN vehicleDetectorOptions2.Table_Row EQUALS OriginalOptions2.	
10	SET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions2.Table_Row IS_EQUAL_TO OriginalOptions2.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.14.17 Configure Paired Detector

Test Procedure:	**Configure Paired Detector**	
Description:	**This test case verifies that the ASC allows a management station to assign a detector to be paired to another detector.**	
Requirement(s):	• **3.5.3.5.3.5 Configure Paired Detector**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDetector** **TestDetector**	**maxVehicleDetectors** **Int** **vehicleDetectorPairedDetector** **vehicleDetectorPairedDetector**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorPairedDetector.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalDetector.	
4	ASSIGN TestDetector EQUALS RANDOM (0 TO MaxRows).	
4.1	IF TestDetector IS_EQUAL_TO OriginalDetector OR TestDetector IS_EQUAL_TO Table_Row.	
4.1.1	GOTO step ‘4’.	
5	ASSIGN vehicleDetectorPairedDetector.Table_Row EQUALS TestDetector.	
6	SET the following objects: vehicleDetectorPairedDetector.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorPairedDetector.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorPairedDetector.Table_Row IS_EQUAL_TO TestDetector.	Pass/Fail
9	ASSIGN vehicleDetectorPairedDetector.Table_Row EQUALS OriginalDetector.	
10	SET the following objects: vehicleDetectorPairedDetector.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorPairedDetector.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorPairedDetector.Table_Row IS_EQUAL_TO OriginalDetector.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.14.18 Configure Paired Detector Placement

Test Procedure:	**Configure Paired Detector Placement**	
Description:	**This test case verifies that the ASC allows a management station to specify if a detector is a leading or trailing detector in a paired placement.**	
Requirement(s):	• **3.5.3.5.3.6 Configure Paired Detector Placement**	
Variable(s):	**MaxRows** **Table_Row** **OriginalOptions2** **TestOptions2**	**maxVehicleDetectors** **Int** **vehicleDetectorOptions2** **vehicleDetectorOptions2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalOptions2.	
4	ASSIGN TestOptions2 EQUALS OriginalOptions2 XOR 2.	
5	ASSIGN vehicleDetectorOptions2.Table_Row EQUALS TestOptions2.	
6	SET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorOptions2.Table_Row IS_EQUAL_TO TestOptions2.	Pass/Fail
9	ASSIGN vehicleDetectorOptions2.Table_Row EQUALS OriginalOptions2.	
10	SET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorOptions2.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorOptions2.Table_Row IS_EQUAL_TO OriginalOptions2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.19 Configure Paired Detector Spacing

Test Procedure:	**Configure Paired Detector Spacing**
Description:	**This test case verifies that the ASC allows a management station to configure a distance between two detectors in a paired placement.**
Requirement(s):	• **3.5.3.5.3.7 Configure Paired Detector Spacing**
Variable(s):	**MaxRows** — **maxVehicleDetectors** **Table_Row** — **Int** **OriginalSpacing** — **vehicleDetectorPairedDetectorSpacing** **TestSpacing** — **vehicleDetectorPairedDetectorSpacing**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorPairedDetectorSpacing.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalSpacing.	
4	ASSIGN TestSpacing EQUALS RANDOM (0 TO 65535).	
	IF TestSpacing IS_EQUAL_TO OriginalSpacing.	
	GOTO step 4.	
5	ASSIGN vehicleDetectorPairedDetectorSpacing.Table_Row EQUALS TestSpacing.	
6	SET the following objects: vehicleDetectorPairedDetectorSpacing.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorPairedDetectorSpacing.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorPairedDetectorSpacing.Table_Row IS_EQUAL_TO TestSpacing.	Pass/Fail
9	ASSIGN vehicleDetectorPairedDetectorSpacing.Table_Row EQUALS OriginalSpacing.	
10	SET the following objects: vehicleDetectorPairedDetectorSpacing.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorPairedDetectorSpacing.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorPairedDetectorSpacing.Table_Row IS_EQUAL_TO OriginalSpacing.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.20 Configure Average Vehicle Length

Test Procedure:	**Configure Average Vehicle Length**	
Description:	**This test case verifies that the ASC allows a management station to store the average vehicle length for a detection zone.**	
Requirement(s):	• **3.5.3.5.3.8 Configure Average Vehicle Length**	
Variable(s):	**MaxRows** **Table_Row** **OriginalLength** **TestLength**	**maxVehicleDetectors** **Int** **vehicleDetectorAvgVehicleLength** **vehicleDetectorAvgVehicleLength**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorAvgVehicleLength.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalLength.	
4	ASSIGN TestLength EQUALS RANDOM (1 TO 4000).	
4.1	IF TestLength IS_EQUAL_TO OriginalLength.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorAvgVehicleLength.Table_Row EQUALS TestLength.	
6	SET the following objects: vehicleDetectorAvgVehicleLength.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorAvgVehicleLength.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorAvgVehicleLength.Table_Row IS_EQUAL_TO TestLength.	Pass/Fail
9	ASSIGN vehicleDetectorAvgVehicleLength.Table_Row EQUALS OriginalLength.	
10	SET the following objects: vehicleDetectorAvgVehicleLength.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorAvgVehicleLength.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorAvgVehicleLength.Table_Row IS_EQUAL_TO OriginalLength.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.14.21 Configure Average Vehicle Length

Test Procedure:	**Configure Vehicle Detection Zone Length**	
Description:	**This test case verifies that the ASC allows a management station to configure the length of a vehicle detection zone.**	
Requirement(s):	• **3.5.3.5.3.9 Configure Vehicle Detection Zone Length**	
Variable(s):	**MaxRows** **Table_Row** **OriginalLength** **Testlength**	**maxVehicleDetectors** **Int** **vehicleDetectorLength** **vehicleDetectorLength**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxVehicleDetectors.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: vehicleDetectorLength.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalLength.	
4	ASSIGN TestLength EQUALS RANDOM (1 TO 65535).	
4.1	IF TestLength IS_EQUAL_TO OriginalLength.	
4.1.1	GOTO step 4.	
5	ASSIGN vehicleDetectorLength.Table_Row EQUALS TestLength.	
6	SET the following objects: vehicleDetectorLength.Table_Row.	Pass/Fail
7	GET the following objects: vehicleDetectorLength.Table_Row.	Pass/Fail
8	VERIFY vehicleDetectorLength.Table_Row IS_EQUAL_TO TestLength.	Pass/Fail
9	ASSIGN vehicleDetectorLength.Table_Row EQUALS OriginalLength.	
10	SET the following objects: vehicleDetectorLength.Table_Row.	Pass/Fail
11	GET the following objects: vehicleDetectorLength.Table_Row.	Pass/Fail
12	VERIFY vehicleDetectorLength.Table_Row IS_EQUAL_TO OriginalLength.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15 Connected Vehicle Interfaces

C.3.15.1 Configure ASC Communications Port for RSU

Test Procedure:	**Configure ASC Communications Port for RSU**	
Description:	**This test case verifies that that ASC allows a management station to configure the physical communications port for exchanging data with an RSU.**	
Requirement(s):	• **3.5.4.1.1 Configure ASC Communications Port for RSU**	
Variable(s):	**OriginalPort** **TestPort** **MaxPorts**	**rsuCommPort** **rsuCommPort** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of physical ports sets the ASC supports'.	
1.1	RECORD this information as MaxPorts.	
2	GET the following objects: rsuCommPort.	Pass/Fail
2.1	RECORD this information as OriginalPort.	
3	ASSIGN TestPort EQUALS RANDOM (0 TO MaxPorts).	
3.1	IF TestPort IS_EQUAL_TO OriginalPort.	
3.1.1	GOTO step 3.	
4	ASSIGN rsuCommPort EQUALS TestPort.	
5	SET the following objects: rsuCommPort.	Pass/Fail
6	GET the following objects: rsuCommPort.	Pass/Fail
7	VERIFY rsuCommPort IS_EQUAL_TO TestPort.	Pass/Fail
8	ASSIGN rsuCommPort EQUALS OriginalPort.	
9	SET the following objects: rsuCommPort.	Pass/Fail
10	GET the following objects: rsuCommPort.	Pass/Fail
11	VERIFY rsuCommPort IS_EQUAL_TO OriginalPort.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.2 Configure Logical RSU Ports and Address

Test Procedure:	**Configure Logical RSU Ports and Address**

Description:	**This test case verifies that the ASC allows a management station to configure an address and port on the RSU for the ASC to send data to.**
Requirement(s):	• **3.5.4.1.2 Configure Logica RSU Ports and Address**
Variable(s):	**MaxRows** **maxRsuPorts** **Table_Row** **Int** **OrignalPortName** **rsuPortName** **OriginalPortNumber** **rsuPortNumber** **TestPortNumber** **rsuPortNumber** **OriginalAddress** **rsuPortAddress**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRsuPorts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: rsuPortName.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPortName.	
4	GET the following objects: rsuPortNumber.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalPortNumber.	
5	GET the following objects: rsuPortAddress.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalAddress.	
6	ASSIGN TestPortNumber EQUALS RANDOM (0 TO 65535)	
6.1	IF TestPortNumber IS_EQUAL_TO OriginalPortNumber	
6.1.1	GOTO step 6.	
7	ASSIGN rsuPortName.Table_Row EQUALS 'Test'.	
8	ASSIGN rsuPortNumber.Table_Row EQUALS TestPortNumber	
9	ASSIGN rsuPortAddress.Table_Row EQUALS '0.0.0.0'.	
10	SET the following objects: rsuPortName.Table_Row, rsuPortNumber.Table_Row. rsuPortAddress.Table_Row.	Pass/Fail
11	GET the following objects: rsuPortName.Table_Row, rsuPortNumber.Table_Row. rsuPortAddress.Table_Row.	Pass/Fail
12	VERIFY rsuPortName.Table_Row IS_EQUAL_TO 'Test'.	Pass/Fail
13	VERIFY rsuPortNumber.Table_Row IS_EQUAL_TO TestPortNumber	Pass/Fail
14	VERIFY rsuPortAddress.Table_Row IS_EQUAL_TO '0.0.0.0'.	Pass/Fail
15	ASSIGN rsuPortName.Table_Row EQUALS OriginalPortName.	
16	ASSIGN rsuPortNumber.Table_Row EQUALS OriginalPortName	
17	ASSIGN rsuPortAddress.Table_Row EQUALS OriginalAddress.	
18	SET the following objects: rsuPortName.Table_Row, rsuPortNumber.Table_Row. rsuPortAddress.Table_Row.	Pass/Fail
19	GET the following objects: rsuPortName.Table_Row, rsuPortNumber.Table_Row. rsuPortAddress.Table_Row.	Pass/Fail
20	VERIFY rsuPortName.Table_Row IS_EQUAL_TO OriginalPortName.	Pass/Fail
21	VERIFY rsuPortNumber.Table_Row IS_EQUAL_TO OriginalPortName	Pass/Fail
22	VERIFY rsuPortAddress.Table_Row IS_EQUAL_TO OriginalAddress.	Pass/Fail
Test Procedure Results		

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.15.3 Configure RSU Interface Polling Period

Test Procedure:	**Configure RSU Interface Polling Period**
Description:	**This test case verifies that the ASC allows a management station to configure the time between polls to an RSU.**
Requirement(s):	• **3.5.4.1.3 Configure RSU Interface Polling Period**
Variable(s):	**MaxRows** maxRsuPorts **Table_Row** Int **OriginalPeriod** rsuPortPollingPeriod **TestPeriod** rsuPortPollingPeriod
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRsuPorts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: rsuPortPollingPeriod.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalPeriod.	
4	ASSIGN TestPeriod EQUALS RANDOM (0 TO 65535)	
4.1	IF TestPeriod IS_EQUAL_TO OriginalPeriod	
4.1.1	GOTO step 4.	
5	ASSIGN rsuPortPollingPeriod.Table_Row EQUALS TestPeriod.	
6	SET the following objects: rsuPortPollingPeriod.Table_Row.	Pass/Fail
7	GET the following objects: rsuPortPollingPeriod.Table_Row.	Pass/Fail
8	VERIFY rsuPortPollingPeriod.Table_Row IS_EQUAL_TO TestPeriod.	Pass/Fail
9	ASSIGN rsuPortPollingPeriod.Table_Row EQUALS OriginalPeriod.	
10	SET the following objects: rsuPortPollingPeriod.Table_Row.	Pass/Fail
11	GET the following objects: rsuPortPollingPeriod.Table_Row.	Pass/Fail
12	VERIFY rsuPortPollingPeriod.Table_Row IS_EQUAL_TO OriginalPeriod.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.4 Configure RSU Interface Watchdog

Test Procedure:	**Configure RSU Interface Watchdog**	
Description:	**This test case verifies that the ASC allows a management station to configure the watchdog time that must elapse whole not activity is detected on a port before a No Activity Fault is reported.**	
Requirement(s):	• **3.5.4.1.4 Configure RSU Interface Watchdog**	
Variable(s):	**MaxRows** **Table_Row** **OriginalTime** **TestTime**	**maxRsuPorts** **Int** **rsuPortWatchdogTime** **rsuPortWatchdogTime**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRsuPorts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: rsuPortWatchdogTime.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalTime.	
4	ASSIGN TestTime EQUALS RANDOM (0 TO 65535).	
4.1	IF TestTime IS_EQUAL_TO OriginalTime	
4.1.1	GOTO step 4.	
5	ASSIGN rsuPortWatchdogTime.Table_Row EQUALS TestTime.	
6	SET the following objects: rsuPortWatchdogTime.Table_Row.	Pass/Fail
7	GET the following objects: rsuPortWatchdogTime.Table_Row.	Pass/Fail
8	VERIFY rsuPortWatchdogTime.Table_Row IS_EQUAL_TO TestTime.	Pass/Fail
9	ASSIGN rsuPortWatchdogTime.Table_Row EQUALS OriginalTime.	
10	SET the following objects: rsuPortWatchdogTime.Table_Row.	Pass/Fail
11	GET the following objects: rsuPortWatchdogTime.Table_Row.	Pass/Fail
12	VERIFY rsuPortWatchdogTime.Table_Row IS_EQUAL_TO OriginalTime.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.5 Monitor RSU Interface Watchdog Timer

Test Procedure:	**Monitor RSU Interface Watchdog Timer**

Description:	**This test case verifies that the ASC allows a management station to view how much time has elapsed since activity was detected on each port.**		
Requirement(s):	• **3.5.4.1.5 Monitor RSU Interface Watchdog Timer**		
Variable(s):	**MaxRows** **Table_Row**	**maxRsuPorts** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRsuPorts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: rsuPortWatchdogTimer.Table_Row	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.6 Enable Signal Phase and Timing Data

Test Procedure:	**Enable Signal Phase and Timing Data**		
Description:	**This test case verifies that that ASC allows a management station to enable or disable SPaT functions.**		
Requirement(s):	• **3.5.4.2.1.1 Enable Signal Phase and Timing Data**		
Variable(s):	**OriginalSpatOptions** **TestSpatOptions**	**spatOptions** **spatOptions**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	GET the following objects: spatOptions.	Pass/Fail
1.1	RECORD this information as OriginalSpatOptions.	
2	ASSIGN TestSpatOptions EQUALS OriginalSpatOptions XOR 1.	
3	ASSIGN spatOptions EQUALS TestSpatOptions.	
4	SET the following objects: spatOptions.	Pass/Fail
5	GET the following objects: spatOptions.	Pass/Fail
6	VERIFY spatOptions IS_EQUAL_TO TestSpatOptions.	Pass/Fail
7	ASSIGN spatOptions EQUALS OriginalSpatOptions.	

8	SET the following objects: spatOptions.	Pass/Fail
9	GET the following objects: spatOptions.	Pass/Fail
10	VERIFY spatOptions IS_EQUAL_TO OriginalSpatOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.7 Retrieve Signal Phase and Timing and Generation Time

Test Procedure:	**Retrieve Signal Phase and Timing Generation Time**
Description:	**This test case verifies that the ASC allows a management station to view the most recent time SPaT data was generated by the ASC.**
Requirement(s):	• **3.5.4.2.1.2 Retrieve Signal Phase and Timing Generation Time**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: spatTimestamp.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.8 Monitor CV Movement Minimum End Time

Test Procedure:	**Monitor CV Movement Minimum End Time**	
Description:	**This test case verifies that the ASC allows a management station to view the earliest possible end time for the current movement state in a connected vehicle environment.**	
Requirement(s):	• **3.5.4.2.1.3.1.1 Monitor Movement Minimum End Time**	
Variable(s):	**MaxSignalGroups Table_Row**	**maxSignalGroups Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxSignalGroups.	
2	FOR Table_Row from 1 to MaxSignalGroups.	
2.1	GET the following objects: signalStateMinEndTick2.Table_Row.1.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.9 Monitor CV Movement Maximum End Time

Test Procedure:	**Monitor CV Movement Maximum End Time**
Description:	**This test case verifies that the ASC allows a management station to view the latest possible end time for the current movement state in a connected vehicle environment.**
Requirement(s):	• **3.5.4.2.1.3.1.2 Monitor Movement Maximum End Time**
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxSignalGroups.	
2	FOR Table_Row from 1 to MaxSignalGroups.	
2.1	GET the following objects: signalStateMaxEndTick2.Table_Row.1.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.10 Monitor CV Movement Expected End Time

Test Procedure:	**Monitor CV Movement Expected End Time**

Description:	**This test case verifies that the ASC allows a management station to view time the current movement state in a connected vehicle environment in expected to end.**		
Requirement(s):	• **3.5.4.2.1.3.1.3 Monitor Movement Likely End Time**		
Variable(s):	**MaxSignalGroups** **Table_Row**	**maxSignalGroups** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: maxSignalGroups.		Pass/Fail
1.1	RECORD this information as MaxSignalGroups.		
2	FOR Table_Row from 1 to MaxSignalGroups.		
2.1	GET the following objects: signalStateLikelyEndTick2.Table_Row.1.		Pass/Fail
Test Procedure Results			
Tested By:		**Date Tested:**	**Pass/Fail**
Test Procedure Notes:			

C.3.15.11 Monitor CV Movement Likely End Time Confidence

Test Procedure:	**Monitor CV Movement Likely End Time Confidence**		
Description:	**This test case verifies that the ASC allows a management station to view the statistical confidence of the expected end time the current movement state in a connected vehicle environment.**		
Requirement(s):	• **3.5.4.2.1.3.1.4 Monitor Movement Likely End Time Confidence**		
Variable(s):	**MaxSignalGroups** **Table_Row**	**maxSignalGroups** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	GET the following objects: maxSignalGroups.		Pass/Fail
1.1	RECORD this information as MaxSignalGroups.		
2	FOR Table_Row from 1 to MaxSignalGroups.		
2.1	GET the following objects: signalStateTickConfidence2.Table_Row.1.		Pass/Fail
Test Procedure Results			

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.15.12 Monitor CV Movement Next Occurrence

Test Procedure:	**Monitor CV Movement Next Occurrence**
Description:	**This test case verifies that the ASC allows a management station to view the expected time the current movement state in a connected vehicle environment will be allowed to proceed again after in ends.**
Requirement(s):	• **3.5.4.2.1.3.1.5 Monitor Movement Next Occurrence**
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxSignalGroups.	
2	FOR Table_Row from 1 to MaxSignalGroups.	
2.1	GET the following objects: signalStateNextTick2.Table_Row.1.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.13 Monitor Next CV Movement Minimum End Time

Test Procedure:	**Monitor Next CV Movement Minimum End Time**
Description:	**This test case verifies that the ASC allows a management station to view the earliest possible end time for movement state after the current movement state in a connected vehicle environment.**
Requirement(s):	• **3.5.4.2.1.3.1.7 Monitor Next Movement Minimum End Time**
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**

Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxSignalGroups.	
2	FOR Table_Row from 1 to MaxSignalGroups.	
2.1	GET the following objects: signalStateMinEndTick2.Table_Row.2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.14 Monitor Next CV Movement Maximum End Time

Test Procedure:	**Monitor Next CV Movement Maximum End Time**	
Description:	**This test case verifies that the ASC allows a management station to view the latest possible end time for movement state after the current movement state in a connected vehicle environment.**	
Requirement(s):	• **3.5.4.2.1.3.1.8 Monitor Next Movement Maximum End Time**	
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxSignalGroups.	
2	FOR Table_Row from 1 to MaxSignalGroups.	
2.1	GET the following objects: signalStateMaxEndTick2.Table_Row.2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.15 Monitor Next CV Movement Maximum Start Time

	Monitor Next CV Movement Maximum Start Time

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to view the start time of the movement state that will follow the current movement state in a connected vehicle environment.**
Requirement(s):	• **3.5.4.2.1.3.1.9 Monitor Next Movement Maximum Start Time**
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxSignalGroups.	
2	FOR Table_Row from 1 to MaxSignalGroups.	
2.1	GET the following objects: signalStateStartTick2.Table_Row.2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.16 Determine Maximum Number of CV Movement Events

Test Procedure:	**Determine Maximum Number of CV Movement Events**
Description:	**This test case verifies that the ASC allows a management station to retrieve the maximum number of movement events in a connected vehicle environment.**
Requirement(s):	• **3.5.4.2.1.3.1.10 Determine Maximum Number of Movement Events**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxMovementEvents.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

Test Procedure Notes:

C.3.15.17 Configure Queue Detectors for CV Movement Assistance

Test Procedure:	**Configure Queue Detectors for CV Movement Assistance**	
Description:	**This test case verifies that the ASC allows a management station to configure the queue detectors for a specific movement in a connected vehicle environment.**	
Requirement(s):	• **3.5.4.1.2.1.3.2.1 Configure Queue Detectors for Movement Assistance**	
Variable(s):	**MaxRows** **Table_Row** **OriginalValue**	**maxMovementManeuvers2** **Int** **movementManeuverQueueDetector2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxMovementManeuvers2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: movementManeuverSignalGroupEntryNumber.Table_Row.	Pass/Fail
4	GET the following objects: movementQueueDetector2.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalValue	
5	ASSIGN movementManeuverQueueDetector2.Table_Row EQUALS '01'.	
6	SET the following objects: movementManeuverQueueDetector2.Table_Row.	Pass/Fail
7	GET the following objects: movementManeuverQueueDetector2.Table_Row.	Pass/Fail
8	VERIFY movementManeuverQueueDetector2.Table_Row IS_EQUAL_TO '01'.	Pass/Fail
9	ASSIGN movementManeuverQueueDetector2.Table_Row EQUALS OriginalValue.	
10	SET the following objects: movementManeuverQueueDetector2.Table_Row.	Pass/Fail
11	GET the following objects: movementManeuverQueueDetector2.Table_Row.	Pass/Fail
12	VERIFY movementManeuverQueueDetector2.Table_Row IS_EQUAL_TO OriginalValue.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.18 Configure Pedestrian Detectors for CV Movement Conflict Assistance

Test Procedure:	**Configure Pedestrian Detectors for CV Movement Conflict Assistance**	
Description:	**This test case verifies that the ASC allows a management station to configure the pedestrian detectors for crossings that may conflict with specific movement in a connected vehicle environment.**	
Requirement(s):	• **3.5.4.1.2.1.3.2.2 Configure Pedestrian Detectors for Movement Conflict Assistance**	
Variable(s):	**MaxRows** **Table_Row** **OriginalDetector**	**maxMovementManeuvers2** **Int** **movementManeuverPedPresense2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxMovementManeuvers2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: movementManeuverSignalGroupEntryNumber.Table_Row.	Pass/Fail
4	GET the following objects: movementManeuverPedPresense2.Table_Row	Pass/Fail
4.1	RECORD this information as OriginalDetector.	
5	ASSIGN movementManeuverPedPresense2.Table_Row EQUALS '01'.	
6	SET the following objects: movementManeuverPedPresense2.Table_Row.	Pass/Fail
7	GET the following objects: movementManeuverPedPresense2.Table_Row.	Pass/Fail
8	VERIFY movementManeuverPedPresense2.Table_Row IS_EQUAL_TO '01'.	Pass/Fail
9	ASSIGN movementManeuverPedPresense2.Table_Row EQUALS OriginalDetector.	
10	SET the following objects: movementManeuverPedPresense2.Table_Row.	Pass/Fail
11	GET the following objects: movementManeuverPedPresense2.Table_Row.	Pass/Fail
12	VERIFY movementManeuverPedPresense2.Table_Row IS_EQUAL_TO OriginalDetector.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.19 Configure Bicycle Detectors for CV Movement Conflict Assistance

Test Procedure:	**15.18 Configure Bicycle Detectors for CV Movement Conflict Assistance**
Description:	**This test case verifies that the ASC allows a management station to configure the bicycle detectors for crossings that may conflict with a specific movement in a connected vehicle environment. Bicycle detectors may be configured as vehicle detectors.**
Requirement(s):	• **3.5.4.1.3.6.3.6 Configure Bicycle Detectors for Movement Assistance V2**
Variable(s):	**MaxRows** **maxMovementManeuvers2** **Table_Row** **Int** **OriginalDetector** **movementManeuverBicyclePresense2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxMovementManeuvers2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: movementManeuverSignalGroupEntryNumber.Table_Row.	Pass/Fail
4	GET the following objects: movementManeuverBicyclePresense2.Table_Row	Pass/Fail
4.1	RECORD this information as OriginalDetector.	
5	ASSIGN movementManeuverBicyclePresense2.Table_Row EQUALS '01'.	
6	SET the following objects: movementManeuverBicyclePresense2.Table_Row.	Pass/Fail
7	GET the following objects: movementManeuverBicyclePresense2.Table_Row.	Pass/Fail
8	VERIFY movementManeuverBicyclePresense2.Table_Row IS_EQUAL_TO '01'.	Pass/Fail
9	ASSIGN movementManeuverBicyclePresense2.Table_Row EQUALS OriginalDetector.	
10	SET the following objects: movementManeuverBicyclePresense2.Table_Row.	Pass/Fail
11	GET the following objects: movementManeuverBicyclePresense2.Table_Row.	Pass/Fail
12	VERIFY movementManeuverBicyclePresense2.Table_Row IS_EQUAL_TO OriginalDetector.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.20 Configure CV Lane Connection Queue Length

Test Procedure:	**Monitor CV Lane Connection Queue Length**
Description:	**This test case verifies that the ASC allows a management station to view the queue length for a specific movement maneuver in a connected vehicle environment.**
Requirement(s):	• **3.5.4.2.1.3.3.1 Monitor Lane Connection Queue Length**
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxMovementManeuvers2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: movementManeuverSignalGroupEntryNumber.Table_Row, movementManeuverQueue2.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.21 Configure CV Lane Connection Vulnerable Road User Detection

Test Procedure:	**Monitor CV Lane Connection Vulnerable Road User Detection**
Description:	**This test case verifies that the ASC allows a management station to view if any pedestrian or bicycle is detected in a connected vehicle environment for a specific movement.**
Requirement(s):	• **3.5.4.2.1.3.3.2 Monitor Lane Connection Vulnerable Road User Detection**
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxMovementManeuvers2.	Pass/Fail

1.1	RECORD this information as MaxRows.	
2	FOR Table_Row from 1 to MaxRows.	
2.1	GET the following objects: movementManeuverSignalGroupEntryNumber.Table_Row, movementManeuverStatus2.Table_Row.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.22 Configure Advisory Speed Type

Test Procedure:	**Configure Advisory Speed Type**
Description:	**This test case verifies that the ASC allows a management station to configure the type of speed advisory for a specific movement traversing a connected intersection.**
Requirement(s):	• **3.5.4.2.1.3.4.1 Configure Advisory Speed Type**
Variable(s):	**MaxRows** **maxAdvisorySpeeds2** **Table_Row** **Int** **OriginalSpeedType** **advisorySpeedType2** **TestSpeedType** **advisorySpeedType2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxAdvisorySpeeds2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: advisorySpeedSignalGroupEntryNumber.Table_Row.	Pass/Fail
4	GET the following objects: advisorySpeedType2.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalSpeedType.	
5	ASSIGN TestSpeedType EQUALS RANDOM (1 TO 4).	
5.1	IF TestSpeedType IS_EQUAL_TO OriginalSpeedType	
5.1.1	GOTO step 5.	
6	ASSIGN advisorySpeedType2.Table_Row EQUALS TestSpeedType.	
7	SET the following objects: advisorySpeedType2.Table_Row.	Pass/Fail
8	GET the following objects: advisorySpeedType2.Table_Row.	Pass/Fail
9	VERIFY advisorySpeedType2.Table_Row IS_EQUAL_TO TestSpeedType.	Pass/Fail
10	ASSIGN advisorySpeedType2.Table_Row EQUALS OriginalSpeedType.	
11	SET the following objects: advisorySpeedType2.Table_Row.	Pass/Fail
12	GET the following objects: advisorySpeedType2.Table_Row.	Pass/Fail

13	VERIFY advisorySpeedType2.Table_Row IS_EQUAL_TO OriginalSpeedType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.23 Configure Advisory Speed

Test Procedure:	**Configure Advisory Speed**	
Description:	**This test case verifies that the ASC allows a management station to configure an advisory speed for a specific movement traversing a connected intersection.**	
Requirement(s):	• **3.5.4.2.1.3.4.2 Configure Advisory Speed**	
Variable(s):	**MaxRows** **Table_Row** **OriginalSpeed** **TestSpeed**	**maxAdvisorySpeeds2** **Int** **advisorySpeedAdvice2** **advisorySpeedAdvice2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxAdvisorySpeeds2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: advisorySpeedSignalGroupEntryNumber.Table_Row.	Pass/Fail
4	GET the following objects: advisorySpeedAdvice2.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalSpeed.	
5	ASSIGN TestSpeed EQUALS RANDOM (0 TO 500).	
5.1	IF TestSpeed IS_EQUAL_TO OriginalSpeed	
5.1.1	GOTO step 5.	
6	ASSIGN advisorySpeedAdvice2.Table_Row EQUALS TestSpeed.	
7	SET the following objects: advisorySpeedAdvice2.Table_Row.	Pass/Fail
8	GET the following objects: advisorySpeedAdvice2.Table_Row.	Pass/Fail
9	VERIFY advisorySpeedAdvice2.Table_Row IS_EQUAL_TO TestSpeed.	Pass/Fail
10	ASSIGN advisorySpeedAdvice2.Table_Row EQUALS OriginalSpeed.	
11	SET the following objects: advisorySpeedAdvice2.Table_Row.	Pass/Fail
12	GET the following objects: advisorySpeedAdvice2.Table_Row.	Pass/Fail
13	VERIFY advisorySpeedAdvice2.Table_Row IS_EQUAL_TO OriginalSpeed.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

Test Procedure Notes:

C.3.15.24 Configure Advisory Speed Zone

Test Procedure:	**Configure Advisory Speed Zone**	
Description:	**This test case verifies that the ASC allows a management station to configure the length of an advisory speed length for a specific movement traversing a connected intersection.**	
Requirement(s):	• **3.5.4.2.1.3.4.3 Configure Advisory Speed Zone**	
Variable(s):	**MaxRows** **Table_Row** **OriginalZoneLength** **TestZoneLength**	**maxAdvisorySpeeds2** **Int** **advisorySpeedZoneLength2** **advisorySpeedZoneLength2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxAdvisorySpeeds2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: advisorySpeedSignalGroupEntryNumber.Table_Row.	Pass/Fail
4	GET the following objects: advisorySpeedZoneLength2.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalZoneLength.	
5	ASSIGN TestZoneLength EQUALS RANDOM (0 TO 10000).	
5.1	IF TestZoneLength IS_EQUAL_TO OriginalZoneLength	
5.1.1	GOTO step 5.	
6	ASSIGN advisorySpeedZoneLength2.Table_Row EQUALS TestZoneLength.	
7	SET the following objects: advisorySpeedZoneLength2.Table_Row.	Pass/Fail
8	GET the following objects: advisorySpeedZoneLength2.Table_Row.	Pass/Fail
9	VERIFY advisorySpeedAdvice2.Table_Row IS_EQUAL_TO TestZoneLength.	Pass/Fail
10	ASSIGN advisorySpeedAdvice2.Table_Row EQUALS OriginalZoneLength.	
11	SET the following objects: advisorySpeedZoneLength2.Table_Row.	Pass/Fail
12	GET the following objects: advisorySpeedZoneLength2.Table_Row.	Pass/Fail
13	VERIFY advisorySpeedAdvice2.Table_Row IS_EQUAL_TO OriginalZoneLength.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.25 Configure Advisory Speed Vehicle Type

Test Procedure:	**Configure Advisory Speed Vehicle Type**
Description:	**This test case verifies that the ASC allows a management station to configure the vehicle type that an advisory speed for a specific movement traversing a connected intersection applies to.**
Requirement(s):	• **3.5.4.2.1.3.4.4 Configure Advisory Speed Vehicle Type**
Variable(s):	**MaxRows** **maxAdvisorySpeeds2** **Table_Row** **Int** **OriginalVehicleClass** **advisorySpeedClass2** **TestVehicleClass** **advisorySpeedClass2**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxAdvisorySpeeds2.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
3	GET the following objects: advisorySpeedSignalGroupEntryNumber.Table_Row.	Pass/Fail
4	GET the following objects: advisorySpeedClass2.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalVehicleClass.	
5	ASSIGN TestVehicleClass EQUALS RANDOM (0 TO 255).	
5.1	IF TestVehicleClass IS_EQUAL_TO OriginalVehicleClass.	
5.1.1	GOTO step 5.	
6	ASSIGN advisorySpeedClass2.Table_Row EQUALS TestVehicleClass.	
7	SET the following objects: advisorySpeedClass2.Table_Row.	Pass/Fail
8	GET the following objects: advisorySpeedClass2.Table_Row.	Pass/Fail
9	VERIFY advisorySpeedClass2.Table_Row IS_EQUAL_TO TestVehicleClass.	Pass/Fail
10	ASSIGN advisorySpeedClass2.Table_Row EQUALS OriginalVehicleClass.	
11	SET the following objects: advisorySpeedClass2.Table_Row.	Pass/Fail
12	GET the following objects: advisorySpeedClass2.Table_Row.	Pass/Fail
13	VERIFY advisorySpeedClass2.Table_Row IS_EQUAL_TO OriginalVehicleClass.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.26 Configure CV Movement States

Test Procedure:	**Monitor CV Movement States**	
Description:	**This test case verifies that the ASC allows a management station to view the current and next movement state in a connected vehicle environment.**	
Requirement(s):	• **3.5.4.2.1.3.5 Monitor Movement State** • **3.5.4.2.1.3.6 Monitor Next Movement State**	
Variable(s):	**MaxSignalGroups** **maxSignalGroups** **Table_Row** **Int**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxSignalGroups.	
2	FOR Table_Row from 1 to MaxSignalGroups.	
2.1	GET the following objects: signalState2.Table_Row.1, signalState2.Table_Row.2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.27 Configure CV Movement Status

Test Procedure:	**Monitor CV Movement Status**	
Description:	**This test case verifies that the ASC allows a management station to view what movements are permitted and when at an intersection with the signal status block in a connected vehicle environment.**	
Requirement(s):	• **3.5.4.2.1.3.7 Monitor Movement Status**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**

1	GET the following objects: signalStatusBlock2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.28 Configure Concurrent Enabled Lanes

Test Procedure:	**Configure Concurrent Enabled Lanes**	
Description:	**This test case verifies that an ASC allows a management station to configure what lanes in a CV environment may be enabled together.**	
Requirement(s):	• **3.5.4.2.1.4.1 Configure Concurrent Enabled Lanes**	
Variable(s):	**MaxLanes** **Table_Row** **OriginalLanes**	**maxEnabledLanesConcurrency** **Int** **enabledLaneConcurrency**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxEnabledLanesConcurrency.	Pass/Fail
1.1	RECORD this information as MaxLanes.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxLanes).	
3	GET the following objects: enabledLaneConcurrency.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalLanes.	
4	ASSIGN enabledLanesConcurrency.Table_Row EQUALS '01'.	
5	SET the following objects: enabledLaneConcurrency.Table_Row.	Pass/Fail
6	GET the following objects: enabledLaneConcurrency.Table_Row.	Pass/Fail
7	VERIFY enabledLaneConcurrency.Table_Row IS_EQUAL_TO '01'.	Pass/Fail
8	ASSIGN enabledLaneConcurrency.Table_Row EQUALS OriginalLanes.	
9	SET the following objects: enabledLaneConcurrency.Table_Row.	Pass/Fail
10	GET the following objects: enabledLaneConcurrency.Table_Row.	Pass/Fail
11	VERIFY enabledLaneConcurrency.Table_Row IS_EQUAL_TO OriginalLanes.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.29 Configure Enabled Lanes by Time of Day

	Configure Enabled Lanes by Time of Day

Test Procedure:	
Description:	**This test case verifies that an ASC allows a management station to configure what lanes in a CV environment may be enabled together.**
Requirement(s):	• **3.5.4.2.1.4.2 Configure Concurrent Enabled Lanes**
Variable(s):	**MaxRows** **maxPatterns** **MaxLanes** **Int** **Table_Row** **Int** **OriginalLanes** **patternSpatEnabledLanes** **TestLanes** **patternSpatEnabledLanes**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	PRE-CONDITION 'The user shall know the maximum number of enabled revocable lane plans supported by the ASC.'	
1.1	RECORD this information as MaxLanes.	
2	GET the following objects: maxPatterns.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows).	
4	GET the following objects: patternSpatEnabledLanes.Table_Row.	Pass/Fail
4.1	RECORD this information as OriginalLanes.	
5	ASSIGN TestLanes EQUALS RANDOM (1 TO MaxLanes).	
5.1	IF TestLanes IS_EQUAL_TO OrignalLanes.	
5.1.1	GOTO step 5.	
6	ASSIGN patternSpatEnabledLanes.Table_Row EQUALS TestLanes.	
7	SET the following objects: patternSpatEnabledLanes.Table_Row.	Pass/Fail
8	GET the following objects: patternSpatEnabledLanes.Table_Row.	Pass/Fail
9	VERIFY patternSpatEnabledLanes.Table_Row IS_EQUAL_TO TestLanes.	Pass/Fail
10	ASSIGN patternSpatEnabledLanes.Table_Row EQUALS OriginalLanes.	
11	SET the following objects: patternSpatEnabledLanes.Table_Row.	Pass/Fail
12	GET the following objects: patternSpatEnabledLanes.Table_Row.	Pass/Fail
13	VERIFY patternSpatEnabledLanes.Table_Row IS_EQUAL_TO OriginalLanes.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.30 Determine CV Lanes Enabled

	Determine CV Lanes Enabled

Test Procedure:	
Description:	**This test case verifies that the ASC allows a management station to view the revocable lanes that are currently active.**
Requirement(s):	• **3.5.4.2.1.4.3 Determine Lanes Enabled**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: spatEnabledLanesStatus.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.31 Enabled Signal Phase and Timing Data Exchange by Port

Test Procedure:	**Enable Signal Phase and Timing Data Exchange by Port**
Description:	**This test case verifies that that ASC allows a management station to enable/disable RSU ports for exchanging SPaT data.**
Requirement(s):	• **3.5.4.2.1.5 Enable Signal Phase and Timing Data Exchange**
Variable(s):	**MaxRows** **maxRsuPorts** **Table_Row** **Int** **OriginalSpatPortOptions** **spatPortOptions** **TestSpatPortOptions** **spatPortOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxRsuPorts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalSpatPortOptions.	
4	ASSIGN TestSpatPortOptions EQUALS OriginalSpatPortOptions XOR 1.	

5	ASSIGN spatPortOptions.Table_Row EQUALS TestSpatPortOptions.	
6	SET the following objects: spatPortOptions.Table_Row.	Pass/Fail
7	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
8	VERIFY spatPortOptions.Table_Row IS_EQUAL_TO TestSpatPortOptions.	Pass/Fail
9	ASSIGN spatPortOptions.Table_Row EQUALS OriginalSpatPortOptions.	
10	SET the following objects: spatPortOptionsTable_Row.	Pass/Fail
11	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
12	VERIFY spatPortOptions.Table_Row IS_EQUAL_TO OriginalSpatPortOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.32 Configure Road Authority Identifier

Test Procedure:	**Configure Road Authority Identifier**		
Description:	**This test case verifies that that ASC allows a management station to configure the agency indicated in SPaT and MAP messages.**		
Requirement(s):	• **3.5.4.2.1.6 Configure Road Authority**		
Variable(s):	**OriginalAuthority** **TestAuthority**	**spatRoadAuthorityID** **spatRoadAuthorityID**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		
Test Step Number	**Test Procedure**		**Results**
1	PRE-CONDITION 'The management station shall be aware of a valid OBJECT IDENTIFIER for the agency'.		
1.1	RECORD this information as TestAuthority.		
2	GET the following objects: spatRoadAuthorityID.		Pass/Fail
2.1	RECORD this information as OriginalAuthority.		
3	ASSIGN spatRoadAuthorityID EQUALS TestAuthority.		
4	SET the following objects: spatRoadAuthorityID.		Pass/Fail
5	GET the following objects: spatRoadAuthorityID.		Pass/Fail
6	VERIFY spatRoadAuthorityID IS_EQUAL_TO TestAuthority.		Pass/Fail
7	ASSIGN spatRoadAuthorityID EQUALS OriginalAuthority.		
8	SET the following objects: spatRoadAuthorityID.		Pass/Fail
9	GET the following objects: spatRoadAuthorityID.		Pass/Fail
10	VERIFY spatRoadAuthorityID IS_EQUAL_TO OriginalAuthority.		Pass/Fail
Test Procedure Results			

Tested By:	Date Tested:	Pass/Fail
Test Procedure Notes:		

C.3.15.33 Retrieve Signal Phase and Timing Intersection Status

Test Procedure:	**Retrieve Signal Phase and Timing Intersection Status**
Description:	**This test case verifies that the ASC allows a management station to identify intersection status alarm indications at a connected intersection. The indications include** • **Manual Control Active** • **Stop Time Active** • **Failure Flash Active** • **Preempt Active** • **Priority Active** • **Fixed Time Active** • **Actuation Active** • **Soft Flash Active** • **Controller Failure** • **Invalid MAP Message** • **Invalid SPaT Data**
Requirement(s):	• **3.5.4.2.1.7.1 Monitor Manual Control Indication** • **3.5.4.2.1.7.2 Monitor Stop Indication** • **3.5.4.2.1.7.3 Monitor Failure Flash Indication** • **3.5.4.2.1.7.4 Monitor Preemption Operation Indication** • **3.5.4.2.1.7.5 Monitor Priority Operation Indication** • **3.5.4.2.1.7.6 Monitor Fixed Time Control** • **3.5.4.2.1.7.7 Monitor Non-Fixed Time Control** • **3.5.4.2.1.7.8 Monitor Standby Operation Indication** • **3.5.4.2.1.7.9 Monitor Controller Failure** • **3.5.4.2.1.7.10 Monitor Map Message Validity** • **3.5.4.2.1.7.11 Monitor SPaT Data Validity**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: spatStatus2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.34 Mark SPaT Invalid - Controller

Test Procedure:	**Mark SPaT Invalid - Controller**		
Description:	**This test case verifies that that ASC allows a management station to mark SPaT data as invalid for the ASC.**		
Requirement(s):	• **3.5.4.2.1.8 Mark SPaT Invalid - Controller**		
Variable(s):	**OriginalSpatOptions** **TestSpatOptions**	**spatOptions** **spatOptions**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**		

Test Step Number	Test Procedure	Results
1	GET the following objects: spatOptions.	Pass/Fail
1.1	RECORD this information as OriginalSpatOptions.	
2	ASSIGN TestSpatOptions EQUALS OriginalSpatOptions XOR 2.	
3	ASSIGN spatOptions EQUALS TestSpatOptions.	
4	SET the following objects: spatOptions.	Pass/Fail
5	GET the following objects: spatOptions.	Pass/Fail
6	VERIFY spatOptions IS_EQUAL_TO TestSpatOptions.	Pass/Fail
7	ASSIGN spatOptions EQUALS OriginalSpatOptions.	
8	SET the following objects: spatOptions.	Pass/Fail
9	GET the following objects: spatOptions.	Pass/Fail
10	VERIFY spatOptions IS_EQUAL_TO OriginalSpatOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.35 Mark SPaT Invalid - Port

Test Procedure:	**Mark SPaT Invalid - Port**	
Description:	**This test case verifies that that ASC allows a management station to mark SPaT data on an RSU port as invalid.**	
Requirement(s):	• **3.5.4.2.1.9 Mark SPaT Invalid - Port**	
Variable(s):	**MaxRows** **Table_Row** **OriginalSpatPortOptions**	**maxRsuPorts** **Int** **spatPortOptions**

	TestSpatPortOptions spatPortOptions	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxRsuPorts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalSpatPortOptions.	
4	ASSIGN TestSpatPortOptions EQUALS OriginalSpatPortOptions XOR 2.	
5	ASSIGN spatPortOptions.Table_Row EQUALS TestSpatPortOptions.	
6	SET the following objects: spatPortOptions.Table_Row.	Pass/Fail
7	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
8	VERIFY spatPortOptions.Table_Row IS_EQUAL_TO TestSpatPortOptions.	Pass/Fail
9	ASSIGN spatPortOptions.Table_Row EQUALS OriginalSpatPortOptions.	
10	SET the following objects: spatPortOptionsTable_Row.	Pass/Fail
11	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
12	VERIFY spatPortOptions.Table_Row IS_EQUAL_TO OriginalSpatPortOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.36 Mark MAP Message Invalid - Controller

Test Procedure:	**Mark MAP Message Invalid - Controller**	
Description:	**This test case verifies that that ASC allows a management station to mark MAP data as invalid for the ASC.**	
Requirement(s):	• **3.5.4.2.1.10 Mark MAP Message Invalid - Controller**	
Variable(s):	**OriginalSpatOptions spatOptions** **TestSpatOptions spatOptions**	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: spatOptions.	Pass/Fail

1.1	RECORD this information as OriginalSpatOptions.	
2	ASSIGN TestSpatOptions EQUALS OriginalSpatOptions XOR 4.	
3	ASSIGN spatOptions EQUALS TestSpatOptions.	
4	SET the following objects: spatOptions.	Pass/Fail
5	GET the following objects: spatOptions.	Pass/Fail
6	VERIFY spatOptions IS_EQUAL_TO TestSpatOptions.	Pass/Fail
7	ASSIGN spatOptions EQUALS OriginalSpatOptions.	
8	SET the following objects: spatOptions.	Pass/Fail
9	GET the following objects: spatOptions.	Pass/Fail
10	VERIFY spatOptions IS_EQUAL_TO OriginalSpatOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.37 Mark MAP Message Invalid – Port

Test Procedure:	**Mark MAP Message Invalid - Port**	
Description:	**This test case verifies that that ASC allows a management station to mark MAP data on an RSU port as invalid.**	
Requirement(s):	• **3.5.4.2.1.11 Mark MAP Message Invalid - Port**	
Variable(s):	**MaxRows** **Table_Row** **OriginalSpatPortOptions** **TestSpatPortOptions**	**maxRsuPorts** **Int** **spatPortOptions** **spatPortOptions**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxRsuPorts.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalSpatPortOptions.	
4	ASSIGN TestSpatPortOptions EQUALS OriginalSpatPortOptions XOR 4.	
5	ASSIGN spatPortOptions.Table_Row EQUALS TestSpatPortOptions.	
6	SET the following objects: spatPortOptions.Table_Row.	Pass/Fail
7	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
8	VERIFY spatPortOptions.Table_Row IS_EQUAL_TO TestSpatPortOptions.	Pass/Fail
9	ASSIGN spatPortOptions.Table_Row EQUALS OriginalSpatPortOptions.	

10	SET the following objects: spatPortOptionsTable_Row.	Pass/Fail
11	GET the following objects: spatPortOptions.Table_Row.	Pass/Fail
12	VERIFY spatPortOptions.Table_Row IS_EQUAL_TO OriginalSpatPortOptions.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.38 Determine Maximum Number of Signal Groups

Test Procedure:	**Determine Maximum Number of Signal Groups**	
Description:	**This test case verifies that the ASC allows a management station to view the number of signal groups supported by the ASC for a connected vehicle environment.**	
Requirement(s):	• **3.5.4.2.1.12.1 Determine Maximum Number of Signal Groups**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxSignalGroups.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.39 Configure Signal Group Intersection Mapping

Test Procedure:	**Configure Signal Group Intersection Mapping**	
Description:	**This test case verifies that that ASC allows a management station to assign signal groups to a connected intersection.**	
Requirement(s):	• **3.5.4.1.2.1.12.2 Configure Signal Group Intersection Mapping**	
Variable(s):	**MaxRows** **Table_Row** **OriginalGroupIntersection** **TestGroupIntersection** **OriginalGroupID**	**maxSignalGroups** **Int** **signalGroupIntersection** **signalGroupIntersection** **signalGroupID**

	TestGroupID signalGroupID
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: signalGroupIntersection.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalGroupIntersection.	
4	ASSIGN TestGroupIntersection EQUALS RANDOM (1 TO 65535)	
4.1	IF TestGroupIntersection IS_EQUAL_TO OrignalGroupIntersection.	
4.1.1	GOTO step 4.	
5	GET the following objects: signalGroupID.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalGroupID.	
6	ASSIGN TestGroupID EQUALS RANDOM (1 TO 255)	
6.1	IF TestGroupID IS_EQUAL_TO OrignalGroupID.	
6.1.1	GOTO step 6.	
7	ASSIGN signalGroupIntersection.Table_Row EQUALS TestGroupIntersection.	
8	ASSIGN signalGroupID.Table_Row EQUALS TestGroupID.	
9	SET the following objects: signalGroupIntersection.Table_Row, signalGroupID.Table_Row.	Pass/Fail
10	GET the following objects: signalGroupIntersection.Table_Row, signalGroupID.Table_Row.	Pass/Fail
11	VERIFY signalGroupIntersection.Table_Row IS_EQUAL_TO TestGroupIntersection.	Pass/Fail
12	VERIFY signalGroupID.Table_Row IS_EQUAL_TO TestGroupID.	Pass/Fail
13	ASSIGN signalGroupIntersection.Table_Row EQUALS OriginalGroupIntersection.	
14	ASSIGN signalGroupID.Table_Row EQUALS OriginalGroupID.	
15	SET the following objects: signalGroupIntersection.Table_Row, signalGroupID.Table_Row.	Pass/Fail
16	GET the following objects: signalGroupIntersection.Table_Row, signalGroupID.Table_Row.	Pass/Fail
17	VERIFY signalGroupIntersection.Table_Row IS_EQUAL_TO OriginalGroupIntersection.	Pass/Fail
18	VERIFY signalGroupID.Table_Row IS_EQUAL_TO OriginalGroupID.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.40 Configure Signal Group Control Source

Test Procedure:	**15.51 Configure Signal Group Control Source**	
Description:	**This test case verifies that that ASC allows a management station to configure the control source (phase or overlap) that controls a signal group in a connected intersection.**	
Requirement(s):	• **3.5.4.1.3.17.3 Configure Signal Group Control Source**	
Variable(s):	**MaxRows** **Table_Row** **MaxPhases** **OriginalControlSource** **TestControlSource** **OriginalControlType** **TestControlType**	**maxSignalGroups** **Int** **maxPhases** **signalGroupControlSource** **signalGroupControlSource** **signalGroupControlType** **signalGroupControlType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: signalGroupID.Table_Row.	Pass/Fail
4	GET the following objects: maxPhases.	Pass/Fail
4.1	RECORD this information as MaxPhases.	
5	GET the following objects: signalGroupControlSource.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalControlSource.	
6	ASSIGN TestControlSource EQUALS RANDOM (1 TO MaxPhases)	
6.1	IF TestControlSource IS_EQUAL_TO OriginalControlSource.	
6.1.1	GOTO step 6.	
7	GET the following objects: signalGroupControlType.Table_Row.	Pass/Fail
7.1	RECORD this information as OriginalControlType.	
8	ASSIGN TestControlType EQUALS RANDOM (2 TO 6)	
8.1	IF TestControlType IS_EQUAL_TO OriginalControlType.	
8.1.1	GOTO step 8.	
9	ASSIGN signalGroupControlSource.Table_Row EQUALS TestControlSource.	
10	ASSIGN signalGroupControlType.Table_Row EQUALS TestControlType.	
11	SET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
12	GET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
13	VERIFY signalGroupControlSource.Table_Row IS_EQUAL_TO TestControlSource.	Pass/Fail

14	VERIFY signalGroupControlType.Table_Row IS_EQUAL_TO TestControlType.	Pass/Fail
15	ASSIGN signalGroupControlSource.Table_Row EQUALS OriginalControlSource.	
16	ASSIGN signalGroupControlType.Table_Row EQUALS OriginalControlType.	
17	SET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
18	GET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
19	VERIFY signalGroupControlSource.Table_Row IS_EQUAL_TO OriginalControlSource.	Pass/Fail
20	VERIFY signalGroupControlType.Table_Row IS_EQUAL_TO OriginalControlType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.41 Configure Signal Group Indication Types

Test Procedure:	**Configure Signal Group Indication Types**	
Description:	**This test case verifies that that ASC allows a management station to configure the type of red and green indications shown in SPaT information messages.**	
Requirement(s):	• **3.5.4.2.1.12.4 Configure Signal Group Indication Types**	
Variable(s):	**MaxRows** **Table_Row** **OriginalGreenType** **TestRedType** **OriginalRedType** **TestRedType**	**maxSignalGroups** **Int** **signalGroupGreenType** **signalGroupGreenType** **signalGroupRedType** **signalGroupRedType**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: signalGroupID.Table_Row.	Pass/Fail
3	GET the following objects: signalGroupGreenType.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalGreenType	
4	ASSIGN TestGreenType EQUALS RANDOM (2 TO 5)	
4.1	IF TestGreenType IS_EQUAL_TO OrignalGreenType.	
4.1.1	GOTO step 4.	

5	GET the following objects: signalGroupRedType.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalRedType.	
6	IF OriginalRedType IS_EQUAL_TO 2	
6.1	ASSIGN TestRedType EQUALS 3	
6.1.1	GOTO step 8.	
7	ASSIGN TestRedType EQUALS 2	
8	ASSIGN signalGroupGreenType.Table_Row EQUALS TestGreenType	
8	ASSIGN signalGroupRedType.Table_Row EQUALS TestRedType.	
9	SET the following objects: signalGroupGreenType, signalGroupRedType.	Pass/Fail
10	GET the following objects: signalGroupGreenType, signalGroupRedType.	Pass/Fail
11	VERIFY signalGroupGreenType.Table_Row IS_EQUAL_TO TestGreenType.	Pass/Fail
12	VERIFY signalGroupRedType.Table_Row IS_EQUAL_TO TestRedType.	Pass/Fail
13	ASSIGN signalGroupGreenType.Table_Row EQUALS OriginalGreenType	
14	ASSIGN signalGroupRedType.Table_Row EQUALS OriginalRedType.	
15	SET the following objects: signalGroupGreenType, signalGroupRedType.	Pass/Fail
16	GET the following objects: signalGroupGreenType, signalGroupRedType.	Pass/Fail
17	VERIFY signaGroupGreenType.Table_Row IS_EQUAL_TO OriginalGreenType.	Pass/Fail
18	VERIFY signalGroupRedType.Table_Row IS_EQUAL_TO OriginalRedType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.42 Configure Signal Group Protected or Permissive State

Test Procedure:	**15.53 Configure Signal Group Protected or Permissive State**	
Description:	**This test case verifies that that ASC allows a management station to configure the control source (phase or overlap) that has secondary control of a signal group in a connected intersection indicating a permissive movement.**	
Requirement(s):	• **3.5.4.2.1.12.5 Configure Signal Group Protected or Permissive State**	
Variable(s):	**MaxRows** **Table_Row** **MaxPhases** **OriginalControlSource** **TestControlSource** **OriginalControlType**	**maxSignalGroups** **Int** **maxPhases** **signalGroupControlSource** **signalGroupControlSource** **signalGroupControlType**

	TestControlType signalGroupControlType
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: maxSignalGroups.	Pass/Fail
1.1	RECORD this information as MaxRows.	
2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: signalGroupID.Table_Row.	Pass/Fail
4	GET the following objects: maxPhases.	Pass/Fail
4.1	RECORD this information as MaxPhases.	
5	GET the following objects: signalGroupControlSource.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalControlSource.	
6	ASSIGN TestControlSource EQUALS RANDOM (1 TO MaxPhases)	
6.1	IF TestControlSource IS_EQUAL_TO OrignalControlSource.	
6.1.1	GOTO step 6.	
7	GET the following objects: signalGroupControlType.Table_Row.	Pass/Fail
7.1	RECORD this information as OriginalControlType.	
8	ASSIGN TestControlType EQUALS RANDOM (2 TO 7)	
8.1	IF TestControlType IS_EQUAL_TO OrignalControlType.	
8.1.1	GOTO step 8.	
9	ASSIGN signalGroupControlSource.Table_Row EQUALS TestControlSource.	
10	ASSIGN signalGroupControlType.Table_Row EQUALS TestControlType.	
11	SET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
12	GET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
13	VERIFY signalGroupControlSource.Table_Row IS_EQUAL_TO TestControlSource.	Pass/Fail
14	VERIFY signalGroupControlType.Table_Row IS_EQUAL_TO TestControlType.	Pass/Fail
15	ASSIGN signalGroupControlSource.Table_Row EQUALS OriginalControlSource.	
16	ASSIGN signalGroupControlType.Table_Row EQUALS OriginalControlType.	
17	SET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
18	GET the following objects: signalGroupControlSource.Table_Row, signalGroupControlType.Table_Row.	Pass/Fail
19	VERIFY signalGroupControlSource.Table_Row IS_EQUAL_TO OriginalControlSource.	Pass/Fail
20	VERIFY signalGroupControlType.Table_Row IS_EQUAL_TO OriginalControlType.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**

Test Procedure Notes:

C.3.15.43 Configure Signal Group Revocable Lanes

Test Procedure:	**Configure Signal Group Revocable Lanes**	
Description:	**This test case verifies that that ASC allows a management station to configure what signal groups may be active based on the revocable lanes that are made active.**	
Requirement(s):	• **3.5.4.1.3.17.6 Configure Signal Group Permissive Lanes**	
Variable(s):	**MaxRows** **Table_Row** **MaxLanes** **OriginalLanes** **TestLanes**	**maxSignalGroups** **Int** **Int** **signalLane** **signalGroupLane**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	

Test Step Number	Test Procedure	Results
1	PRE-CONDITION 'The user shall know the maximum number of enabled revocable lane plans supported by the ASC.'	
1.1	RECORD this information as MaxLanes.	
2	GET the following objects: maxSignalGroups.	Pass/Fail
2.1	RECORD this information as MaxRows.	
3	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
4	GET the following objects: signalGroupID.Table_Row.	Pass/Fail
5	GET the following objects: signalGroupControLane.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalLanes.	
6	ASSIGN TestControlSource EQUALS RANDOM (1 TO MaxLanes)	
6.1	IF TestLanes IS_EQUAL_TO OrignalLanes.	
6.1.1	GOTO step 6.	
7	ASSIGN signalGroupLane.Table_Row EQUALS TestLanes.	
8	SET the following objects: signalGroupLane.Table_Row.	Pass/Fail
10	GET the following objects: signalGroupLane.Table_Row.	Pass/Fail
11	VERIFY signalGroupLane.Table_Row IS_EQUAL_TO TestLanes.	Pass/Fail
12	ASSIGN signalGroupLane.Table_Row EQUALS OriginalLane.	
13	SET the following objects: signalGroupLane.Table_Row.	Pass/Fail
14	GET the following objects: signalGroupLane.Table_Row.	Pass/Fail
15	VERIFY signalGroupLane.Table_Row IS_EQUAL_TO OriginalLanes.	Pass/Fail

Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.44 Determine Maximum Number of Signal State Entries

Test Procedure:	**Determine Maximum Number of Signal State Entries**
Description:	**This test case verifies that the ASC allows a management station to view the number of signal states that may be configured.**
Requirement(s):	• **3.5.4.1.2.1.12.7 Determine Maximum Number of Signal Groups**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxAgencySignalStates.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.45 Determine Maximum Number of Signal State Entries

Test Procedure:	**Configure Signal State Parameters**
Description:	**This test case verifies that that ASC allows a management station to configure what signal states may be indicated in SPaT information messages dependent on signal color and permissive movements.**
Requirement(s):	• **3.5.4.1.3.17.8 Configure Signal State Parameters**
Variable(s):	**MaxRows** **maxSignalGroups** **Table_Row** **Int** **OriginalColor** **agencySignalStateColor** **OriginalControlType** **agencySignalStateControlType** **OriginalStateOptions** **agencySignalStateOptions** **OriginalStateValue** **agencySignalStateValue**
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: maxAgencySignalStates.	Pass/Fail
1.1	RECORD this information as MaxRows.	

2	ASSIGN Table_Row EQUALS RANDOM (1 TO MaxRows)	
3	GET the following objects: agencySignalStateColor.Table_Row.	Pass/Fail
3.1	RECORD this information as OriginalColor	
4	GET the following objects: agencySignalStateControlType.Table_Row	Pass/Fail
4.1	RECORD this information as OriginalControlType.	
5	GET the following objects: agencySignalStateOptions.Table_Row.	Pass/Fail
5.1	RECORD this information as OriginalStateOptions.	
6	GET the following objects: agencySignalStateValue.Table_Row.	Pass/Fail
6.1	RECORD this information as OriginalStateValue.	
7	ASSIGN agencySignalStateColor.Table_Row EQUALS 2.	
7.1	NOTE '2 = red'.	
8	ASSIGN agencyStateControlType.Table_Row EQUALS 2.	
8.1	NOTE '2 = vehicle'.	
9	ASSIGN agencySignalStateOptions.Table_Row EQUALS 272	
10	ASSIGN agencySignalStateValue.Table_Row EQUALS 5	
10.1	NOTE '2 = stopAndRemain'.	
11	SET the following objects: agencySignalStateColor.Table_Row, agencySignalStateControlType.Table_Row, agencySignalStateOptions.Table_Row, agencySignalStateValue.Table_Row.	Pass/Fail
12	GET the following objects: agencySignalStateColor.Table_Row, agencySignalStateControlType.Table_Row, agencySignalStateOptions.Table_Row, agencySignalStateValue.Table_Row.	Pass/Fail
13	VERIFY agencySignalStateColor.Table_Row IS_EQUAL_TO 2.	Pass/Fail
13	VERIFY agencySignalStateControlType.Table_Row IS_EQUAL_TO 2.	Pass/Fail
14	VERIFY agencySignalStateOptions.Table_Row IS_EQUAL_TO 272.	Pass/Fail
15	VERIFY agencySignalStateValue.Table_Row IS_EQUAL_TO 5.	Pass/Fail
16	ASSIGN agencySignalStateColor.Table_Row EQUALS OriginalColor.	
17	ASSIGN agencyStateControlType.Table_Row EQUALS OriginalControlType.	
18	ASSIGN agencySignalStateOptions.Table_Row EQUALS OriginalStateOptions	
19	ASSIGN agencySignalStateValue.Table_Row EQUAIS OriginalStateValue.	
20	SET the following objects: agencySignalStateColor.Table_Row, agencySignalStateControlType.Table_Row, agencySignalStateOptions.Table_Row, agencySignalStateValue.Table_Row.	Pass/Fail
21	GET the following objects: agencySignalStateColor.Table_Row, agencySignalStateControlType.Table_Row, agencySignalStateOptions.Table_Row, agencySignalStateValue.Table_Row.	Pass/Fail
23	VERIFY agencySignalStateColor.Table_Row IS_EQUAL_TO OriginalColor.	Pass/Fail
24	VERIFY agencySignalStateControlType.Table_Row IS_EQUAL_TO OrignalControlType.	Pass/Fail
25	VERIFY agencySignalStateOptions.Table_Row IS_EQUAL_TO OriginalStateOptions.	Pass/Fail

26	VERIFY agencySignalStateValue.Table_Row IS_EQUAL_TO OriginalStateValue.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.46 Retrieve Signal Phase and Timing Time Point

Test Procedure:	**Retrieve Signal Phase and Timing Time Point**	
Description:	**This test case verifies that the ASC allows a management station to view movement current time on the ASC used as the reference for SPaT data generation. This time may differ from the time on an RSU.**	
Requirement(s):	• **3.5.4.1.3.18 Retrieve Signal Phase and Timing Time Point V2**	
Variable(s):		
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**	
Test Step Number	**Test Procedure**	**Results**
1	GET the following objects: ascCurrentTick2.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		

C.3.15.47 Retrieve MAP Plan in Effect

Test Procedure:	**Retrieve MAP Plan in Effect**
Description:	**This test case verifies that the ASC allows a management station to view the MAP plan currently being broadcasted in a connected vehicle environment.**
Requirement(s):	• **3.5.4.3.3.1 Retrieve MAP Plan in Effect**
Variable(s):	
Pass/Fail Criteria:	**The device under test shall pass every verification step in this test case to pass the test case.**

Test Step Number	Test Procedure	Results
1	GET the following objects: mapActivatePlan.	Pass/Fail
Test Procedure Results		
Tested By:	**Date Tested:**	**Pass/Fail**
Test Procedure Notes:		