

National Aeronautics and Space  
Administration

Ames Research Center  
Moffett Field, California 94035

# **Live Virtual Constructive (LVC)**

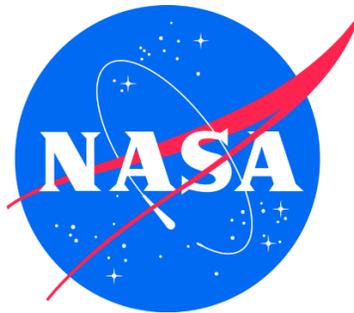
## **Interface Control Document For the LVC Gateway**

Document #: **LVC ICD-03**

Release: **Rev F**

January 19, 2017

**UAS-NAS**  
**Live Virtual Constructive – Distributed Environment (LVC)**  
**Message Interface Control Document**  
**For the LVC Gateway**



Prepared by

Srba Jovic, Software Manager

Science Applications International Corporation (SAIC)  
NASA Ames Research Center  
Moffett Field, CA 94035-0081

Prepared By:

---

Srba Jovic – IT&E - LVC SW & Integration, SAIC/NASA ARC

Concur:

---

Jamie Willhite – IT&E - LVC Integration & Test, NASA AFRC

---

Jack Connolly – IT&E Systems Engineer, MEI/NASA ARC

---

Neil Otto – Test Operations Lead, SAIC/NASA ARC

---

Jim Murphy – IT&E Co-Technical Lead, NASA ARC

---

Sam Kim – IT&E Co-Technical Lead, NASA AFRC

Approve:

---

Heather Maliska – SPM, NASA AFRC

## Record of Changes

<i>Version #</i>	<i>Date</i>	<i>Page #</i>	<i>Author</i>	<i>Description</i>
Baseline	May 1, 2013	All	Srba Jovic	Initial Release of Document
Rev A	Mar 28, 2014		Srba Jovic	Updates for IHITL
Rev B	Feb 26, 2015	All	Srba Jovic	Update for FT3 Requirements, including addition of Stratway+ SS Band messages and Omni Band messages.
Rev C	Sept 21, 2015	All	Srba Jovic	Update for PT6 Requirements, including addition of TCAS message as used at ARC
Rev D	Jan 05, 2016	All	Srba Jovic	Added TCAS message as used at LaRC
Rev E	April 04, 2016	All	Srba Jovic	Added Flight State Noisy message in support of Part Task 6 experiment. The new message is generated after the truth state data is passed through the Honeywell sensor model.
Rev F	January 19, 2017	All	Srba Jovic	Added new messages are in support of ACAS-Xu FT2 Experiment. Added AC Track State Messages for ownship and intruder, and ACAS-Xu alerting and guidance message. Also, previous data structures are converted into table format.

## Table of Contents

1	Introduction .....	8
2	Applicable Documents .....	9
2.1	References .....	9
2.2	Standards .....	9
3	Known LVC System Clients .....	10
4	Definition of Messages Used in the LVC System .....	10
4.1	General Message Construct .....	10
4.2	Message Types .....	11
4.3	Message Structures .....	14
4.3.1	Handshake Data Structure .....	14
4.3.2	MPI Aircraft Flight State Data Structure .....	18
4.3.3	DAA Aircraft Track State Data Structure for the Ownship .....	21
4.3.4	DAA Aircraft Track State Data Structure for Intruders .....	25
4.3.5	MPI Aircraft Flight Plan Structure .....	29
4.3.6	Aircraft Flight Trajectory Intent Structure .....	32
4.3.7	Aircraft Delete Structure .....	37
4.3.8	Set Ownship Structure .....	37
4.3.9	Sense and Avoid (SAA) Threat Results Message .....	38
4.3.10	SAA Release Structure .....	42
4.3.11	Sense and Avoid (SAA) Resolution Maneuver Structure .....	42
4.3.12	Sense and Avoid (SAA) Resolution Reroute Structure .....	45
4.3.13	Navigation Mode Message Structure .....	46
4.3.14	ACAS_Xu Data Structures .....	48
4.3.15	Stratway Bands Data Structure .....	54
4.3.16	Omni (SAA) Band Message .....	56
4.3.17	TCAS 7.0 Message Structures .....	57
4.3.18	TCAS 7.1 Message Structures .....	62
4.3.19	Well Clear Recovery Message .....	65
4.4	Message Content Notes .....	67
4.4.1	Heartbeat Message .....	67
4.4.2	Primitive Data Type Definitions and Sizes in Bytes .....	67
4.4.3	Byte Order and Need for Byte Swapping .....	68
5	Acronym List .....	69
6	Appendix A. Detect and Avoid Symbology .....	72
7	Appendix B. ADRS Proc and Category Values .....	73

## List of Figures

<b>Figure 1.</b> General System Architecture for UAS-NAS Baseline LVC Simulation .....	8
--	---

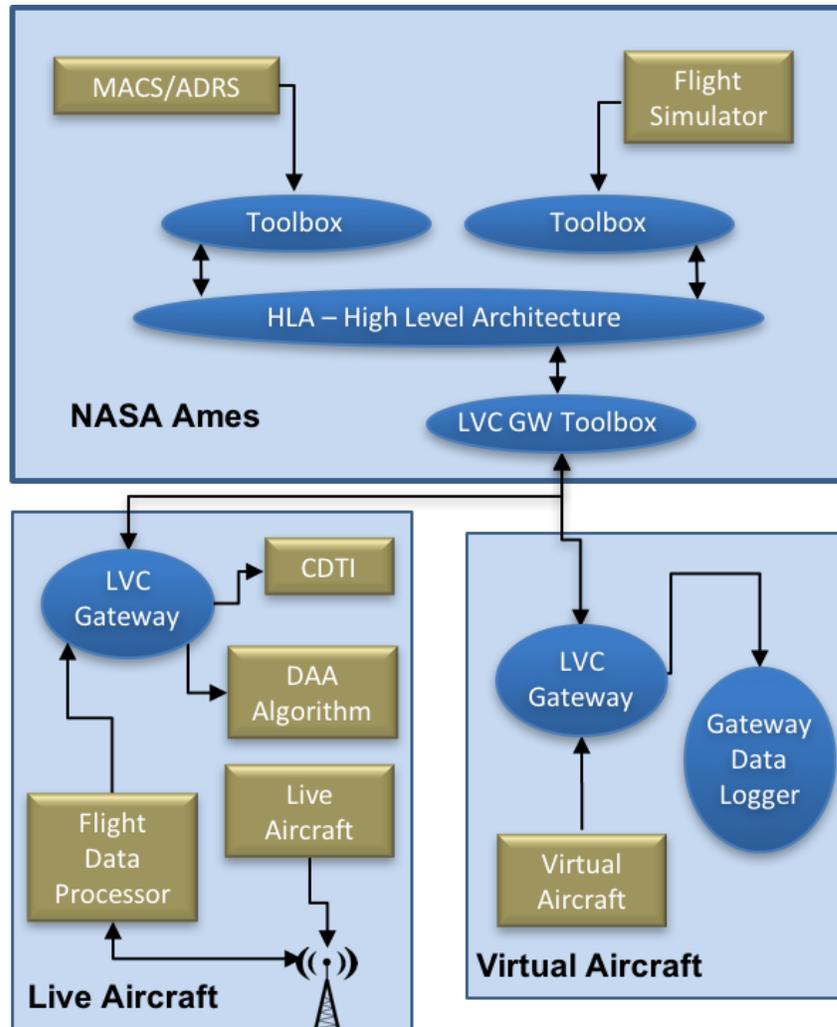
## List of Tables

Table 1. Source Id .....	10
Table 2. Message Header .....	10
Table 3. Message Type Id .....	11
Table 4. Handshake Message .....	15
Table 5. MPI Flight State Message .....	18
Table 6. DAA Aircraft Track State Data Structure for the Ownship .....	22
Table 7. DDA Aircraft Track State Data Structure for Intruders .....	26
Table 8. Flight Plan Message .....	29
Table 9. Trajectory Intent Message .....	32
Table 10. Trajectory Specification Structure .....	33
Table 11. Trajectory Waypoint Structure .....	34
Table 12. Trajectory Waypoint Types .....	36
Table 13. Delete Aircraft Message .....	37
Table 14. Set Ownship Message .....	37
Table 15. SAA Threat Results Message .....	38
Table 16. SAA Threat Specification Structure .....	38
Table 17. SAA Threat Message .....	39
Table 18. Alert Levels, Definitions .....	42
Table 19. SAA Release Message .....	42
Table 20. SAA Resolution Maneuver Message .....	42
Table 21. SAA Resolution Maneuver Specification .....	43
Table 22. SAA Maneuver .....	43
Table 23. SAA Maneuver Type .....	43
Table 24. SAA Resolution Reroute Message .....	45
Table 25. SAA Resolution Reroute Specifications .....	45
Table 26. SAA Resolution Waypoint Structure .....	46
Table 27. Navigation Mode Message .....	47
Table 28. Navigation Mode .....	48
Table 29. ACAS-Xu Alerting and Guidance Message .....	48
Table 30. Vertical Resolution Advisory - Combined Control .....	50
Table 31. Vertical Resolution Advisory - Vertical Control .....	50
Table 32. Vertical Resolution Advisory - Up Advisory .....	51
Table 33. Vertical Resolution Advisory - Down Advisory .....	51
Table 34. Horizontal Resolution Advisory - Combined Control .....	51
Table 35. Horizontal Resolution Advisory Aural Alerts .....	52
Table 36. Vertical Resolution Advisory Interpretation .....	53
Table 37. TCAS Alert Levels .....	54
Table 38. Stratway Interval Type .....	54
Table 39. Stratway Interval .....	55
Table 40. Stratway Intruder .....	55
Table 41. Stratway Bands Message .....	55
Table 42. Omni Bands Interval .....	57
Table 43. Omni Bands Message .....	57
Table 44. TCAS 7.0 Alert Structure .....	58
Table 45. TCAS 7.0 Threat Table .....	60
Table 46. TCAS 7.0 Aural Action Table .....	60

Table 47. TCAS 7.0 Combined Control .....	61
Table 48. TCAS 7.0 Vertical Control .....	61
Table 49. TCAS 7.0 Up Advisory .....	61
Table 50. TCAS 7.0 Down Advisory.....	62
Table 51. RA Output to Display and Aural Subsystem Every Cycle .....	62
Table 52. TCAS 7.1 Message .....	63
Table 53. TCAS 7.1 Intruder State .....	63
Table 54. TCAS 7.1 Alert Level.....	63
Table 55. TCAS 7.1 Aural Alert.....	64
Table 56. TCAS 7.1 Combined Control .....	64
Table 57. TCAS 7.1 Down Advisory.....	65
Table 58. TCAS 7.1 Up Advisory .....	65
Table 59. Well Clear Recovery Message.....	66
Table 60. Directional Guidance .....	66
Table 61. Guidance Type.....	66
Table 62. Range Type .....	66
Table 63. Range .....	67

## 1 Introduction

This Interface Control Document (ICD) contains the requisite information needed by any component of Live Virtual and Constructive (LVC) system to connect and communicate through the LVC Gateway. The information in this document represents the complete set of protocols for all Project research requirements and objectives. The purpose of this ICD is to provide users of the LVC Gateway with the information to develop software that can connect to, and send, and receive data through the LVC Gateway. It should be noted up front that the LVC Software Design Description (LVC SWDD-03) is a companion to the ICD and should be used to complement the details described here.



**Figure 1.** General System Architecture for UAS-NAS Baseline LVC Simulation

The general system architecture, shown in Figure 1, details network connectivity between distributed subsystem components for the live, virtual, and constructive integrated testing and evaluation in support of the UAS-in-the-NAS Project. LVC system components are classified into the two general categories – LVC Core Components, shown as blue nodes, and LVC Participants, shown as brown nodes in Figure 1. LVC Core Components are made up of:

1. The LVC Gateway, which is middleware developed by the NASA Ames IT&E team, and
2. Pitch RTI (industry HLA IEEE 1516 standard), which is middleware infrastructure with the suite of High Level Architecture (HLA) Toolboxes supporting specific interface requirements imposed by each participant.
3. HLA Toolboxes, used to connect participating systems to the LVC
4. LVC Data Logger, used to record all messages handled by an instance of the LVC Gateway.

LVC Participants are what provide and use the data passing through the LVC. Examples include:

1. Target generators providing live, virtual and constructive traffic
2. Ground Control Station (GCS) simulators
3. Air Traffic Control (ATC) environment(s)
4. Pilot Traffic Displays
5. Detect and Avoid (DAA) alerting and guidance software

The integrated LVC environment connects participating clients through the HLA and the LVC Gateway through Toolbox component(s) and software wrappers that translate the internal software data into the messaging scheme outlined in this document.

## 2 Applicable Documents

The following documents (or later, earlier versions superseded) form a part of this document to the extent specified herein. In the event of conflict between the documents referenced herein and the contents of this document, the contents of this document shall be considered a superseding requirement.

### 2.1 References

LVC SWRD-02 Rev D	LVC Software Requirements Document
LVC SRD-01 Rev D	LVC System Requirements Document
RGCS SRD-01 Rev B	Research Ground Control Station
LVC SWDD-03 Rev C	LVC Software Description Document

### 2.2 Standards

TCP/IP	Transmission Control Protocol / Internet Protocol (IPV4 /IPV6)
--------	--

### 3 Known LVC System Clients

Table 1 provides a list of known client systems that can connect to the LVC System. The ID from the table is a unique number that identifies that client as a specific type of client. In order to add a new type of client, either one of the existing IDs must be used or a new ID must be created.

**Table 1. Source Id**

Source/Client Name	Id	Description
CSD	101	Deprecated
IkhanaSim	102	Ikhana Simulator
IkhanaUAS	103	Ikhana Aircraft
LangleyUAS	104	Langley's UAS Aircraft
LVCDataLogger	105	LVC Data Logger
UASRP	106	
LVCGateway	107	LVC Gateway
GlennUAS	108	Glenn's UAS Aircraft
ADRS	109	MACS generates Tracks
SaaProc	111	Sense And Avoid Processor – JADEM wrapper
VSCS	112	Vigilant Spirit Control Station
CPDS/IOServer	113	Interface to the live data source
StratwayGCS/ DAIDALUS	114	The name Stratway has been replaced by DAIDALUS
ACASXU	115	ACAS-Xu provides horizontal and vertical CA and DAA alerting and guidance
ExelisNextGen	116	Exelis (Harris) provides live data for NAS
TCAS	117	TCAS Processor. Contains TCAS II v7.1
SSAM	118	Sensor model adds noise to truth track state data
FPG	119	Flight Plan Generator

### 4 Definition of Messages Used in the LVC System

#### 4.1 General Message Construct

Every Multi-Purpose Interface (MPI) protocol message exchanged between the LVC Gateway and a client system will have a header immediately followed by the payload of the corresponding message.

**Table 2. Message Header**

Data Type	Data Field Name	Description
int	MsgTypeId	Defined by Message Type in MsgTypeId Table
int	MsgSize	Header size + payload size defined by each message structure
int	SrcId	Defined by Source/Client Name in Source Id Table

The message type defined in the header indicates the type of message contained in the payload. Total size of the header structure is 12 bytes.

#### 4.2 Message Types

Table 3 identifies different message types that are supported by the LVC Gateway and maps them to their respective data structures. Message data structure of corresponding message types are defined in the next section.

Note that the new terminology for Sense and Avoid (SAA) has been introduced into the Phase 1 DAA MOPS. SAA has been renamed to Detect and Avoid (DAA). However, it has been decided to retain all the legacy references to SAA in all of the pertinent messages in this ICD. This preserves and maintains consistent terminology between the current ICD and the software that has been developed using the previous versions of the ICD for the earlier phases of the UAS-in-the-NAS project.

**Table 3. Message Type Id**

Message Type	ID	Message Structure	Description
MsgFlightState	5310	MPI Aircraft Flight State Data Structure	Contains specific in-flight information for a target in the LVC as defined in the Flight State Messages table.
MsgFlightPlan	5201	MPI Aircraft Flight Plan Structure	Contains specific Flight Plan information for a target in the LVC as defined in the Flight Plan Messages table.
MsgTrajectoryIntent	5421	MPI Aircraft Flight Trajectory Intent Structure	Contains information on planned future aircraft behavior in the LVC as defined in the Trajectory Intent Messages structure.
MsgDeleteAc	5202	Aircraft Delete Structure	A message initiated by LVC Gateway clients, which drop an aircraft from the simulation.
MsgHandshake	5960	Handshake Data Structure	A message responsible for initiating the connection between the client and the server. The message registers the client with the server and sets up a publish/subscribe dependency.
MsgSetOwnship	5901	Set Ownship Structure	Sets the call sign of the ownship.
MsgFlightStateADSB	7010	MPI Aircraft Flight State Data Structure	If the UAS is equipped with the ADS-B "In" capability, then the client will publish the ADS_B state data of the surrounding traffic including its own.
MsgFlightStateTISB	7011	MPI Aircraft Flight State Data Structure	If the UAS is equipped with the TIS-B "In" capability, then the client will publish the TIS_B state data of the surrounding traffic including its own.
MsgHeartbeat	7030		An LVC Gateway message sent to every client at a configurable time interval for the sole purpose of detecting whether the client socket port has been closed.

Message Type	ID	Message Structure	Description
MsgSaaThreatResults	5830	SAA Threat Results Message Structure	Contains information on the number, type, and severity of intruder aircraft as defined in the SAA Threat Messages table.
MsgSaaResManeuvers	5831	MsgSaaResManeuverSt	Contains information on the number and type of resolution maneuvers as defined in the SAA Maneuver Messages table.
MsgSaaResReroute	5832	MsgSaaResRerouteSt	Contains information on SAA reroute maneuvers as defined in the SAA Reroute Messages table.
MsgSaaFlightState [1]	5833	MPI Aircraft Flight State Data Structure	A message resulting from the sensor surveillance range filtering as part of the sensor model applied to the entire simulated traffic. Required to display traffic on the ownship display. The SAA Aircraft Flight State message is a result of the sensor surveillance range filtering (part of the sensor model) applied to the all traffic defined by the MsgFlightState message. The filtered traffic is rebroadcast as a MsgSaaFlightState message. (This message may soon be deprecated)
MsgSaaRelease	5834	SAA Release Structure	Message that indicates that a conflict has been cleared as a result of executing a previously advised maneuver.
MsgNavMode	5835	Navigation Mode Message Structure	Used whenever the ownship flight control system executes a maneuver or when the ownship consumes a waypoint on the route to send the SAA system intent information.

Message Type	ID	Message Structure	Description
MsgTrialTrajectoryIntent	5454	Aircraft Flight Trajectory Intent Structure	Used during trial planning assessment of maneuvers or waypoints by the ownship. Sent to SaaProc component for conflict assessment. Trial planning messages are sent at a 15Hz rate to the SaaProc component for conflict assessment with intruders.
MsgSaaTrialThreatResults	5839	MsgSaaResManeuverSt	Contains information provided by the SAA algorithm on the number, type, and severity of intruder aircraft based on information provided in the Trial Trajectory Intent message.
MsgSaaRecapManeuver	5840	MsgSaaResRerouteSt	To be determined
MsgTrialAccepted	5452		Not used
MsgStrwayBands	5841	Stratway Bands Data Structure	Provides Stratway+ bands data from the DAIDALUS system to subscribing clients.
MsgAcasxu	5842	ACAS_Xu Data Structure	Provides CA and DAA alerting and guidance information generated by the ACAS-Xu algorithm
MsgSaaBands	5843	Omni (SAA) Band Message	Provides SAA bands data from the SAA/JADEM Process system to subscribing clients.
MsgTcasAlert	5850	TCAS 7.0 Alert Message Structure	Provides Resolution Advisory and Traffic Advisory Information generated by the TCAS 7.0 LVC system component
MsgWellClearRecovery	5851	Well Clear Recovery Message Structure	Provides limited suggestive guidance with guidance options in a particular direction, e.g. left turn, right turn, climb, and descend. Inhibited during an active TCAS RA.
MsgLaRCTcasAlert	5852	TCAS 7.1 Message Structure	Provides Resolution Advisory and Traffic Advisory Information generated by the TCAS 7.1 LVC system component

Message Type	ID	Message Structure	Description
MsgAcTrackStateOST	5854	DAA Aircraft Track State Data Structure for the Ownship	Aircraft Track State for the ownship containing TRUTH data. This data type conforms to the MOPS SC-228 document.
MsgAcTrackStateT	5855	DAA Aircraft Track State Data Structure for Intruders	Aircraft Track State for intruders containing TRUTH data. This data type conforms to the MOPS SC-228 document.
MsgAcTrackStateOS [2]	5856	DAA Aircraft Track State Data Structure for the Ownship	Aircraft Track State for the ownship containing live or processed data. This data type conforms to the MOPS SC-228 document.
MsgAcTrackState [3]	5857	DAA Aircraft Track State Data Structure for Intruders	Aircraft Track State for intruders containing live or processed data. This data type conforms to the MOPS SC-228 document.

Notes:

[1] The SAA Aircraft Flight State message is a result of the sensor surveillance range filtering (part of the sensor model) applied to the all traffic defined by the MsgFlightState message. The filtered traffic is rebroadcast as a MsgSaaFlightState message. The SAA Aircraft Flight State structure, **MsgSaaFlightState**, is a duplicate of MsgFlightState in Table 5.

[2] The Aircraft Track State message containing noisy data (labeled as MsgACTrackStateOS) is a result of either the modeled surveillance sensor noise or provided by the live surveillance sensors for the ownship. The Aircraft Track State structure, **MsgACTrackStateOS**, is a duplicate in its form of MsgTrackStateOST defined in Table 6.

[3] The Aircraft Track State message containing noisy data (labeled as MsgACTrackState) is a result of either the modeled surveillance sensor noise or provided by the live surveillance sensors for the ownship. The Aircraft Track State structure, **MsgACTrackState**, is a duplicate in its form of MsgTrackStateOST defined in Table 7.

### 4.3 Message Structures

This section contains the definition of each of the message type data structures.

#### 4.3.1 Handshake Data Structure

The Handshake Data structure is defined below. This structure represents the payload of a message sent by the client upon establishing the connection with the LVC Gateway (the server). The client handshake message header contains Source parameters that define the data source identification and names as defined in Table 4.

**Table 4. Handshake Message**

<b>Data Type</b>	<b>Data Field Name</b>	<b>Description</b>
char	clientName[12]	Client that is connecting to the LVC gateway defined by Source/Client Name in SrcId Table
char	dataProviderName[12]	Callsign if the ownship, otherwise empty string
bool	b_publish_MsgFlightState	Sets up a publish dependency for the Flight State Message
bool	b_publish_MsgFlightPlan	Sets up a publish dependency for the Flight Plan Message
bool	b_publish_MsgTrajectory	Sets up a publish dependency for the Trajectory Message
bool	b_publish_MsgFlightStateADSB	Sets up a publish dependency for the ADS-B Flight State Message
bool	b_publish_MsgFlightStateTISB	Sets up a publish dependency for the TIS-B Flight State Message
bool	b_subscribe_MsgFlightState	Sets up a subscribe dependency for the Flight State Message
bool	b_subscribe_MsgFlightPlan	Sets up a subscribe dependency for the Flight Plan Message
bool	b_subscribe_MsgTrajectory	Sets up a subscribe dependency for the Trajectory Message
bool	b_subscribe_MsgFlightStateADSB	Sets up a subscribe dependency for the ADS-B Flight State Message
bool	b_subscribe_MsgFlightStateTISB	Sets up a subscribe dependency for the TIS-B Flight State Message
bool	b_publish_MsgDeleteAc	Sets up a publish dependency for the Delete Aircraft Message
bool	b_subscribe_MsgDeleteAc	Sets up a subscribe dependency for the Delete Aircraft Message
bool	b_publish_MsgSetOwnship	Sets up a publish dependency for the Set Ownship Message
bool	b_subscribe_MsgSetOwnship	Sets up a subscribe dependency for the Set Ownship Message
bool	b_publish_MsgSaaFlightState	Sets up a publish dependency for the SAA Flight State Message
bool	b_subscribe_MsgSaaFlightState	Sets up a subscribe dependency for the SAA Flight State Message
bool	b_publish_MsgSaaThreatResults	Sets up a publish dependency for the SAA Threat Results Message
bool	b_subscribe_MsgSaaThreatResults	Sets up a subscribe dependency for the SAA Threat Results Message
bool	b_publish_MsgSaaResManeuver	Sets up a publish dependency for the SAA Resolution Maneuver Message
bool	b_subscribe_MsgSaaResManeuver	Sets up a subscribe dependency for the SAA Resolution Maneuver Message
bool	b_publish_MsgSaaResReroute	Sets up a publish dependency for the SAA Resolution Reroute Message
bool	b_subscribe_MsgSaaResReroute	Sets up a subscribe dependency for the SAA Resolution Reroute Message
bool	b_publish_MsgSaaRelease	Sets up a publish dependency for the SAA Release Message

Data Type	Data Field Name	Description
bool	b_subscribe_MsgSaaRelease	Sets up a subscribe dependency for the SAA Release Message
bool	b_publish_MsgTrialTrailIntent	Sets up a publish dependency for the Trial Trail Intent Message
bool	b_subscribe_MsgTrialTrailIntent	Sets up a subscribe dependency for the Trial Trail Intent Message
bool	b_publish_MsgSaaTrialThreatResults	Sets up a publish dependency for the Saa Trial Threat Results Message
bool	b_subscribe_MsgSaaTrialThreatResults	Sets up a subscribe dependency for the Saa Trial Threat Results Message
bool	b_publish_MsgSaaTrialRecapManeuver	Sets up a publish dependency for the Saa Trial Recap Maneuver Message
bool	b_subscribe_MsgSaaTrialRecapManeuver	Sets up a subscribe dependency for the Saa Trial Recap Maneuver Message
bool	b_publish_MsgTrialAccepted	Sets up a publish dependency for the Trial Accepted Message
bool	b_subscribe_MsgTrialAccepted	Sets up a subscribe dependency for the Trial Accepted Message
bool	b_publish_MsgNavigationMode	Sets up a publish dependency for the Navigation Mode Message
bool	b_subscribe_MsgNavigationMode	Sets up a subscribe dependency for the Navigation Mode Message
bool	b_publish_MsgSaaBands	Sets up a publish dependency for the SAA Bands Message
bool	b_subscribe_MsgSaaBands	Sets up a subscribe dependency for the SAA Bands Message
bool	b_publish_MsgAcasxu	Sets up a publish dependency for the ACAS-Xu Message
bool	b_subscribe_MsgAcasxu	Sets up a subscribe dependency for the ACAS-Xu Message
bool	b_publish_MsgStrwayBands	Sets up a publish dependency for the Stratway Bands Message
bool	b_subscribe_MsgStrwayBands	Sets up a subscribe dependency for the Stratway Bands Message
bool	b_publish_MsgTcasAlert	Sets up a publish dependency for the TCAS 7.0 alerting and guidance message
bool	b_subscribe_MsgTcasAlert	Sets up a subscribe dependency for the TCAS 7.0 CA alert and guidance message
bool	b_publish_MsgWellClearRecovery	Sets up a publish dependency for the Well Clear Recovery Message
bool	b_subscribe_MsgWellClearRecovery	Sets up a subscribe dependency for the Well Clear Recovery Message
bool	b_publish_MsgLaRCTcasAlert	Sets up a publish dependency for the TCAS 7.1 CA alerting and guidance message
bool	b_subscribe_MsgLaRCTcasAlert	Sets up a subscribe dependency for the TCAS 7.1 CA alerting and guidance message
bool	b_publish_MsgTCASSurveillance	Set up a publish dependency for the TCAS Surveillance message
bool	b_subscribe_MsgTCASSurveillance	Set up a subscribe dependency for the TCAS Surveillance message

Data Type	Data Field Name	Description
bool	b_publish_MsgAcTrackStateOST	Sets up a publish dependency for the AcTrackStateOST for truth Surveillance Message for the ownship
bool	b_subscribe_MsgAcTrackStateOST	Sets up a subscribe dependency for the AcTrackStateOST for truth Surveillance Message for the ownship
bool	b_publish_MsgAcTrackStateT	Sets up a publish dependency for the AcTrackStateT truth Surveillance Message for intruders
bool	b_subscribe_MsgAcTrackStateT	Sets up a subscribe dependency for the AcTrackStateT truth Surveillance Message for intruders
bool	b_publish_MsgAcTrackStateOS	Sets up a publish dependency for the AcTrackStateOST Surveillance Message for the ownship
bool	b_subscribe_MsgAcTrackStateOS	Sets up a subscribe dependency for the AcTrackStateOST Surveillance Message for the ownship
bool	b_publish_MsgAcTrackState	Sets up a publish dependency for the AcTrackState Surveillance Message for intruders
bool	b_subscribe_MsgAcTrackState	Sets up a subscribe dependency for the AcTrackState Surveillance Message for intruders

The role of the handshake message is twofold: 1) it is responsible for initiating the connection between the client and the server; 2) it registers the client with the server and sets up a publish/subscribe dependency.

For example, if the client that connects to LVC Gateway is VSCS, then the clientName is “VSCS” as defined in Table 1. The dataProviderName attribute is set to an empty string. The construct with the dataProviderName field is deprecated as the Cockpit Situation Display (CSD) participant has been replaced with VSCS. The data field is maintained for back-compatibility in case the CSD is introduced back into the LVC system. In that case, the dataProviderName attribute is set to the call sign of the ownship that provides ownship data for the CSD client.

The client can publish and subscribe to certain data types specified by the Boolean attributes in the structure defined above. If the client is a VSCS entity then the first three Booleans (b\_publish\_MsgFlightState, b\_publish\_MsgFlightPlan, and b\_publish\_MsgTrajectory) will be set to “true” indicating to the LVC Gateway server that the client will publish its own flight state vector, flight plan and trajectory intent.

If the UAS is equipped with the ADS-B ”In” capability then the Boolean attribute b\_publish\_MsgFlightStateADSB will be set to “true” indicating that the client will publish the ADS\_B state data of surrounding traffic. In that case, the flag m\_equipageFlags in the MsgFlightState structure should be set by the publishing client to a value as defined in Table 5. Note that m\_equipageFlags is set to a zero for all other Flight State messages that are not generated using ADS-B and/or TIS-B tracks.

The Ikhana Sim will not consume external data. Hence all subscribe attributes will be set to false, indicating to the server that it should not send any traffic data to the Ikhana Sim client.

### 4.3.2 MPI Aircraft Flight State Data Structure

The Aircraft Flight State structure is defined below. This structure represents the payload of an aircraft flight state message.

Note that some test environments do not generate some of the data fields defined in the message. Those values should be set to either -999999 for integers, -999999.0 for floats and doubles depending upon the variable type.

Data fields represented by strings will be published with the constant length as defined in the message interface. If a string is shorter than the allocated space, blank spaces should be filled with “\0” (a null character). For example, “AAL123” should be represented as “AAL123\0\0\0\0\0” in a 12 character array.

**Table 5. MPI Flight State Message**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
char	m_acid [eMPI_ID_LENGTH=12]				Aircraft Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_cid				Computer Identification	Unique numerical identifier for each aircraft.
double	m_timeCreated	UTC Decimal seconds			Time when the Flight State Message was created.	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
double	m_timeReceived	UTC Decimal seconds			Time when the Flight State Message was received.	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
double	m_latitude	Decimal degrees	-90 <sup>o</sup>	+90 <sup>o</sup>	Latitude of aircraft	Northern latitudes are positive, Southern latitudes are negative. Expressed to 6 decimal places. Example: 43.068286
double	m_longitude	Decimal degrees	-180 <sup>o</sup>	+180 <sup>o</sup>	Longitude of aircraft	Eastern longitudes are positive, Western longitudes are negative. Expressed to 6 decimal places. Example: 43.068286
float	m_pressure Altitude	Feet			Indicated altitude of aircraft above mean sea level on a standard day.	
float	m_geoAltitude	Feet			Altitude of aircraft above or below the WGS-84 ellipsoid.	Not supported

Data Type	Data Field Name	Units	Min	Max	Description	Notes
float	m_indicatedAirSpeed	Knots			Airspeed read directly from the airspeed indicator on an aircraft, driven by the pitot-static system	
float	m_mach				The speed of the aircraft related to the speed of sound.	Expressed as number with 6 decimal places. Example 0.23458
float	m_bankAngle	Degrees	-90 <sup>o</sup>	+90 <sup>o</sup>	The angle at which the vehicle is inclined about its longitudinal axis with respect to the horizontal.	Left bank expressed as negative; right bank expressed as positive.
float	m_pitchAngle	Degrees	-90 <sup>o</sup>	+90 <sup>o</sup>	The angle between the aircraft's longitudinal axis and the horizontal plane.	Descent expressed as negative; climb expressed as positive.
float	m_groundSpeed	Knots			Aircraft's speed relative to the ground.	Expressed as number with 6 decimal places. Example 130.23458
float	m_verticalSpeed	Feet per minute (fpm)			The rate of climb or descent of an aircraft indicated in feet per minute.	Expressed as number with 6 decimal places. Example 130.23458
float	m_trueHeading	Degrees	0 <sup>o</sup>	360 <sup>o</sup>	Aircraft heading in relation to true North.	Expressed as number with 6 decimal places. Example 130.23458
float	m_magneticVariation	Degrees	0 <sup>o</sup>	180 <sup>o</sup>	The horizontal angle between the true north and magnetic north measured east or west.	West is expressed as a negative. East is expressed as a positive.
float	m_trueGroundTrack	Degrees	0 <sup>o</sup>	360 <sup>o</sup>	The path on the surface of the Earth directly below the aircraft as true heading.	A heading expressed as number with 6 decimal places. Example 130.23458
float	m_trueAirSpeed	Knots			The speed of the aircraft relative to the air mass in which it is flying.	Expressed as number with 6 decimal places. Example 130.23458
float	m_altitudeTarget	Feet			The altitude toward which the aircraft is directed at target generation.	
float	m_headingTarget	Degrees	0 <sup>o</sup>	360 <sup>o</sup>	The magnetic heading toward which the aircraft is directed at target generation.	A heading expressed as number with 6 decimal places. Example 130.23458
float	m_speedTarget	Knots			The airspeed toward which the aircraft is directed at target generation.	Expressed as number with 6 decimal places. Example 130.23458

Data Type	Data Field Name	Units	Min	Max	Description	Notes
float	m_verticalSpeed Target	Feet per minute (fpm)			The rate of climb or descent toward which the aircraft is directed at target generation.	Expressed as number with 6 decimal places. Example 130.23458. Not supported.
int	m_equipageFlags		0	72	Denotes the type of airborne sensors aboard the aircraft	Sensor source is represented by an 8 bit bitwise word designating surveillance sensors and their combination in providing track data. 0: TCAS Active Surveillance Target 1: ADS-B (1090MHz) Target 2: ADS-B (978MHz) Target 3: TIS-B Target 4: Reserved 5: Passive Non-Cooperative Sensor (i.e. EO/IR) 6: Active Non-Cooperative Sensor (i.e. Radar) 7: RESERVED Example: Active Surveillance and Radar are on (bit values are set to 1) – the equipage flag value is equal to 65.
int	m_modeFlags				Denotes the flight status of the aircraft	Value displayed a series of status denotations. Example: 2101256
int	m_dlinkFlags				Denotes the data link capability of the aircraft	Not supported
int	m_configurationFlags					Not supported
float	m_flaps	Degrees	0	30	Denotes position of the aircraft's flaps	
float	m_speedBreaks				Denotes status of aircraft's speed brakes	Not supported
float	m_windDirection	Degrees	0 <sup>0</sup>	360 <sup>0</sup>	Wind direction	Expressed as number with 6 decimal places. Example 130.23458. Wind direction frame of reference depends on the data sources.
float	m_windSpeed	Knots			Wind speed	Expressed as number with 6 decimal places. Example 130.23458
float	m_outerAir Temperature	Fahrenheit			Air temperature	Not supported

Data Type	Data Field Name	Units	Min	Max	Description	Notes
float	m_mapRangeCaptain	NM	10	320	MAP range for PIC (Pilot in command)	Not supported
float	m_mapRangeFo	NM	10	320	MAP range for FO (First Officer)	Not supported
float	m_headingBug	Degrees	0 <sup>0</sup>	360 <sup>0</sup>	Heading bug on MAP	Not supported
float	m_vhfFrequency				Assigned ATC frequency	Not supported
int	m_beaconCode		0000	7777	4096 Transponder code	Octal value for each digit.
int	m_geoSectorId				Geodetic position of aircraft.	Not supported
int	m_atcSectorId				ATC center sector in which the aircraft is located.	Center designator_numeric value of sector. Example: ZOA_41 (Center) or ZOA_236 (Approach)
int	m_acSectorId[ eMPI_STRING_SECTOR=8 ]				ATC approach control sector in which the aircraft is located.	Center designator_numeric value of sector. Example: ZOA_41 (Center) or ZOA_236 (Approach)
char	m_atcSectorName				Name of the ATC sector in which the aircraft is located.	Center designator_numeric value of sector. Example: ZOA_41 (Center) or ZOA_236 (Approach)
int	m_ICAO_Code					Used to transmit ICAO code as there is no dedicated field for that attribute in any of the predefined structures.

*Note: The entity that is equipped with ADS-B will publish flight data that maps to the MsgFlightState structure with the m\_equipageFlag field set to the value specified above that corresponds to the ADS-B data.*

*Note: The m\_modeFlags field is set at least to ADRS\_MPI\_FS\_LNAV (value of 1) in order for trajectory intent to show in the MACS.*

#### 4.3.3 DAA Aircraft Track State Data Structure for the Ownship

The Aircraft Track State data structure for the ownship is defined in Table 6. This structure represents the payload of an aircraft track state message that conforms to the RTCA SC-228 DAA Phase 1 MOPS document (with additional data fields used for internal message tracking). It is the baseline structure for the MsgACTrackStateOS and MsgACTrackStateOST messages.

Data fields represented by strings will be published with the constant length as defined in the message interface. If a string is shorter than the allocated space, blank spaces should be filled with “\0” (a null

character). For example, “AAL123” should be represented as “AAL123\0\0\0\0\0” in a 12 character array.

**Table 6. DAA Aircraft Track State Data Structure for the Ownship**

Data Type	Fieldname	Units	Min	Max	Description
char	callsign[MAX_CALLSIGN_LEN=12]	Non-dimensional			Aircraft callsign
int	uid	Non-dimensional			Unique Track Id
double	timeOfApplicability	Decimal seconds			Positive value representing Unix Epoch Time. Value in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places. Example: 1474041713.363 (representing 16 Sep 2016 16:01:52 UTC).  DAA MOPS Note 1
double	timeReceived	Decimal seconds			Positive value representing Unix Epoch Time. Value in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places. Example: 1474041713.363 (representing 16 Sep 2016 16:01:52 UTC)
double	timeSent	Decimal seconds			Positive value representing Unix Epoch Time. Value in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places. Example: 1474041713.363 (representing 16 Sep 2016 16:01:52 UTC)
double	Latitude	Decimal degrees	-90	90	DAA MOPS Notes 1,2  Signed latitude in decimal degrees format, +North/-South. Specify the angle with 7 positions following decimal point. It is based on reference to WGS-84.

Data Type	Fieldname	Units	Min	Max	Description
double	Longitude	Decimal degrees	-180	180	DAA MOPS Notes 1,2 Signed longitude in decimal degrees format. +East/-West. Specify 7 positions following decimal point. It is based on reference to WGS-84.
int	horizontalPosition Uncertainty	Non-dimensional	0	13	DAA MOPS Note 3 The 95% Accuracy Figure of Merit for Horizontal Position (HFOM). The 95% bound of horizontal position uncertainty exceeds 0.1 NM (NACp < 7).  0: Estimated Position Uncertainty $\geq$ 18.52 km ( $\geq$ 10 nm) 1: EPU < 18.52 km (10 nm) 2: EPU < 7.4 km (4 nm) 3: EPU < 3.7 km (2 nm) 4: EPU < 1852 m (1 nm) 5: EPU < 926 m (0.5 nm) 6: EPU < 555.6 m (0.3 nm) 7: EPU < 185.2 m (0.1 nm) 8: EPU < 92.6 m (0.05 nm) 9: EPU < 30m 10: EPU < 10m 11: EPU < 3m
int	horizontalPosition Integrity	Non-dimensional	0	15	DAA MOPS Note 3 0: Radius of Containment is unknown; 1: <20 NM (37.04 km); 2: <8 NM (14.816 km); 3: <4 NM (7.408 km); 4: <2 NM (3.704 km); 5: <1 NM (1852 m); 6: <0.6 NM (1111 m); 7: <0.5 NM (926 m); 8: <0.3 NM (555 m); 9: <0.2 NM (370m); 10: <0.1 NM (185.2 m); 11: < 75 m; 12: < 25 m; 13: < 7.5 m
int	horizontalVelocityType	Non-dimensional	1	2	1=NS/EW, 2=Magnitude/ Track angle
float	horizontalVelocity_1	knots			This field is either +N/-S, or horizontal vector magnitude (ground speed)
float	horizontalVelocity_2	knots/decimal degrees			This field is either +E/-W, or true track angle.
float	horizontalVelocity Uncertainty-1	int	0	4	0:Horizontal Velocity Error us Unknown or $\geq$ 10m/s 1: < 10m/s; 2: < 3m/s; 3: < 1m/s; 4: <0.3m/s

Data Type	Fieldname	Units	Min	Max	Description
float	horizontalVelocity Uncertainty-2	knots/decimal degrees			Represents corresponding accuracy of the Horizontal Velocity 2 data field
int	geometricAltitude	Feet			The height of aircraft center of mass above the WGS84 ellipsoid as measured using the GNSS. Note: provided by DTIF
int	geometricAltitude Uncertainty	Feet			Note 1,3
float	verticalVelocity	Feet/min			N/A
float	verticalVelocity Uncertainty	Feet/min			
int	pressureAltitude	Feet			Pressure altitude is the (uncorrected) altitude relative to the standard sea-level pressure reference [29.92" inches of Mercury/1013.25 hectopascals (hPa)]. Traffic pressure altitude is the pressure altitude transmitted by a target.
int	ICAO_Address	Non-dimensional			ICAO Assigned Mode S transponder 24-bit address
int	isAirborneStatus	Non-dimensional	FALSE	TRUE	Aircraft status indicates if th AC is in air or not. DAA MOPS Note 5: Airborne/Ground state determination on UAS may require algorithms based on inputs from multiple sensors to accurately detect the airborne/ground status. The A/G state should be the same state as determined by the on-board transponder to report airborne/on-the-ground state per RTCA DO-181.
float	trueHeading	Decimal degrees	0	359	DAA MOPS TABLE 2-3 Note 6: The horizontal velocity accuracy performance may be established by analysis of the installed equipment, and then will need to be evaluated during the certification/installation process.

Data Type	Fieldname	Units	Min	Max	Description
float	rollAngle	Decimal degrees	-180	180	Aircraft roll angle where positive value is roll right (left wing up).
float	pitchAngle	Decimal degrees			Aircraft pitch angle where positive value is pitch up.
float	roll Rate	Decimal degrees per second			Aircraft roll rate
float	pitch Rate	Decimal degrees per second			Aircraft pitch rate
float	headingRate	Decimal degrees per second			Aircraft turn rate
int	validityInformation	Non-dimensional			DAA MOPS Note 9: Indication of failure, degradation or validation of the sources will be sent to the DAA processor.
float	trueAirSpeed	knots			True air speed. Note: Not in Table 2-3 for DAA MOPS Ownship Data. Provided by DTIF Interface. Introduced by SSI.

*Note: Some test environments do not generate some of the data fields defined in the message those values should be set to either -999999 for integers, -999999 for floats and doubles, or to an empty string initialized to a NULL, depending upon the variable type.*

#### 4.3.4 DAA Aircraft Track State Data Structure for Intruders

The Aircraft Track State data structure for the intruder is defined in Table 7. This structure represents the payload of an intruder aircraft track state message that conforms to the RTCA SC-228 DAA Phase 1 MOPS document (with additional data fields used for internal message tracking). It is the baseline structure for the MsgACTrackState and MsgACTrackStateT messages.

Data fields represented by strings will be published with the constant length as defined in the message interface. If a string is shorter than the allocated space, blank spaces should be filled with “\0” (a null character). For example, “AAL123” should be represented as “AAL123\0\0\0\0\0” in a 12 character array.

Table 7. DDA Aircraft Track State Data Structure for Intruders

Data Type	Data Fieldname	Units	Min	Max	Description
char	callsign[MAX_CALLSIGN_LEN]	Non-dimensional			Aircraft callsign or tail number
int	uid	Non-dimensional			Unique Track ID
double	timeOfApplicability	Decimal seconds			<p>Time of applicability (TOA) value represents Unix Epoch Time. Value in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places. Example: 1474041713.363 (representing 16 Sep 2016 16:01:52 UTC)</p> <p>Note: TOA is provided by CPDS/IOserver ( UTC button on the CPDS Bridge UI should be engaged by the user).</p>
double	timeReceived	Decimal seconds			<p>Positive value representing Unix Epoch Time. Value in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places. Example: 1474041713.363 (representing 16 Sep 2016 16:01:52 UTC)</p>
double	timeSent	Decimal seconds			<p>Positive value representing Unix Epoch Time. Value in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places. Example: 1474041713.363 (representing 16 Sep 2016 16:01:52 UTC)</p>
int	intruderAlertStatus	Non-dimensional			<p>Non Threat Traffic, Traffic Advisory, Resolution Advisory, Proximate Traffic</p> <p>NOTE: provided by DTIF - lines 97-100.</p>

Data Type	Data Fieldname	Units	Min	Max	Description
int	dataSourceType	Non-dimensional	0	72	<p>ADS-B 1090ES (Air-Air); Active Surveillance TCAS; Air-to-Air Radar; (Reserved) ADS-B UAT (Air-to-Air); (Reserved) Automatic Dependent Surveillance- Rebroadcast (ADS-R); (Reserved) TIS-B; (Reserved) Passive Non- Cooperative Sensor;</p> <p>DTIF DISPmatrix: Active Surveillance Target (TCAS); ADS-B sensor (1090MHz); ADS-B sensor (978MHz); TIS-B Target; Passive Non-Cooperative (EO/IR); Active Non - Cooperative (Radar);</p> <p>Note: Above data is provided by DTIF DISPmatrix - lines 89-95.</p>
int	validatedADSB	Non-dimensional			Validated, Unvalidated, or Previously Validated
int	isAirborneStatus	Non-dimensional			An indication of whether an intruder is on the ground or in the air helps to differentiate and de-clutter the traffic display. Only ADS-B will have a direct indication of whether the traffic is on the ground ("on-ground") or in the air; otherwise the aircraft is assumed to be in airborne mode.
int	coordinateType	Non-dimensional			1=Relative Lat/Relative Long 2=Range/Bearing 3=Lat/Long
double	horizontalPosition_1	Dimensions are specified according to coordinate type.			This field maps to either Latitude, Relative Latitude, or Range

Data Type	Data Fieldname	Units	Min	Max	Description
double	horizontalPosition_2	Dimensions are specified according to coordinate type.			This field maps to either Longitude, Relative Longitude, or Bearing
float	horizontalPosition Accuracy	Dimension is specified according to coordinate type.			Horizontal Position Accuracy correlates to the corresponding coordinate type provided above
int	bearingInvalid	Non-dimensional			Data field is invalid
int	horizontalVelocity Type	Non-dimensional			1=NS/EW, 2=Horizontal Velocity Magnitude/Track angle
float	horizontalVelocity_1	Knots			This field is either NS, or Horizontal Velocity vector magnitude
float	horizontalVelocity_2	Knots			This field is either EW, or true track angle
float	horizVelocity Accuracy-1	Knots			Represents corresponding accuracy of the Horizontal Velocity 1 data field
float	horizVelocity Accuracy-2	Knots			Represents corresponding accuracy of the Horizontal Velocity 2 data field
int	altitudeType	Non-dimensional			1=pressureAltitude, 2=geometricAltitude, 3=relativePressAltitude, 4=relativeGeomAltitude
float	Altitude	Feet			Elevation of intruder from ownship to traffic.
int	relativeAltitude Accuracy	Non-dimensional			TBD
int	altitudeInvalid	Non-dimensional	FALSE	TRUE	
int	verticalSpeedType	Non-dimensional			1=absoluteVerticalSpeed 2=relativeVerticalSpeed
float	verticalSpeed	Feet/min			This field is either absoluteVertical Speed, or relativeVerticalSpeed
float	verticalSpeed Accuracy	Feet/min			This field is either absoluteVerticalSpeedAccuracy, or relativeVerticalSpeedAccuracy

*Note: Some test environments do not generate some of the data fields defined in the message those values should be set to either -999999 for integers, -999999 for floats and doubles, or to an empty string initialized to a NULL, depending upon the variable type.*

### 4.3.5 MPI Aircraft Flight Plan Structure

The Aircraft Flight Plan structure is defined below. This structure represents the payload of an aircraft flight plan message. All messages displayed below are defined in the adrs\_mpi.h interface used ADRS.

**Table 8. Flight Plan Message**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
int	m_dataSource				Defines where the Flight Plan message originates	
char	m_acid [eMPI_ID_LENGTH=12]				Aircraft Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_adrsProc				Defines the MACS internal process that denotes where a message originated from.	Values may be accessed in Appendix B, with a value of 38 to indicate a VAST connection.
int	m_cid				Computer Identification	Unique numerical identifier for each aircraft.
char	m_type[ eMPI_STRING_TYP E=16 ]				Manufacture and body type. Used to populate flight information such as Line 4 in the FDB.	
char	m_gateName[ eMPI_STRING_ TYPE=20 ]				Name of the gate assigned to the aircraft.	
char	m_meterFixName[ eMPI_STRING_ TYPE=20 ]				Name of a fix along an established route from over which aircraft will be metered prior to entering terminal airspace.	
char	m_outerFixName[ eMPI_STRING_ TYPE=20 ]				Name of an adapted fix along the converted route of flight, prior to the meter fix, for which crossing times are calculated and displayed in the metering position list.	
int	m_category				Defines MACS aircraft departure/arrival/overflight category.	Values may be accessed in Appendix B Table B1

Data Type	Data Field Name	Units	Min	Max	Description	Notes
char	m_route[eMPI_STRING_FILED_ROUTE=300]				The defined path, consisting of one or more courses in a horizontal plane, which aircraft traverse over the surface of the earth.	Used by MACS to generate the flight path of the aircraft.
char	m_departureFix[eMPI_ID_LENGTH=12 ]				Point of origin for the flight plan generated in MACS.	
int	m_departureTime	UTC	0000	2359	Time when the aircraft leaves the departure fix.	
int	m_assignedAltitude	Feet			The altitude/ flight level assigned by ATC and acknowledged by the pilot.	
float	m_filedSpeed	Knots			The airspeed filed by the pilot. Normally refers to KIAS.	
int	m_timeEnroute	hhmmss			The flying time from departure point to destination.	
float	m_approachSpeed	Knots			The speed of the aircraft when making an approach to landing.	
float	m_landingSpeed	Knots			The speed of the aircraft after landing.	
char	m_coordinationFrd[eMPI_STRING_NAME=20 ]				The fix, stated as fix, radial, and distance, in relation to which facilities will handoff, transfer control of an aircraft, or coordinate flight progress data.	
char	m_coordinationFix[eMPI_STRING_NAME=20 ]				The fix in relation to which facilities will handoff, transfer control of an aircraft, or coordinate flight progress data.	May be stated as Latitude/longitude or by Fix name.
float	m_coordnationX				The fix in relation to which facilities will handoff, transfer control of an aircraft, or coordinate flight progress data as defined as a horizontal point on the computer display.	

Data Type	Data Field Name	Units	Min	Max	Description	Notes
float	m_coordinationY				The fix in relation to which facilities will handoff, transfer control of an aircraft, or coordinate flight progress data as defined as a vertical point on the computer display.	
int	m_faaCoordTime	hhmmss			The time prior to the coordination fix where ATC coordination is affected.	Lead in times for features such as auto-handoff or auto-display may be set for the test sectors in MACS.
int	m_coordinationTime	UTC			The time when an aircraft crosses a coordination fix.	
char	m_destinationFix[eMPI_STRING_NAME=20 ]				Destination airport or the fix where the route terminates.	
char	m_destinationName[eMPI_STRING_NAME=20 ]				Name of the destination airport or the fix where the route terminates.	
char	m_runwayName[eMPI_STRING_NAME=20 ]				Name of the runway at the destination airport.	Example: SFO_28L
int	m_configuration					
int	m_beacon				The 4096 selectable Mode 3/A aircraft transponder code.	
char	m_acType[eMPI_STRING_NAME=16 ]				Manufacture and body type. Used by MACS to populate the flight characteristics of the aircraft.	
int	m_timeReceived	UTC Deci mal secon ds			Time when the Flight State Message was created.	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
short	m_status				Defines the status of a flight in MACS	1=proposed flight 2=estimated flight 3=depart flight 4=blocked slot
char	m_fpDataSource				Defines where the Flight Plan message originates	
char	m_equipment Available					
int	m_dlinkEquipped				Defines aircraft ability to receive and send ATC datalink messages.	

This is an example of a `m_route` field in the Flight Plan structure conforming to the standard FAA syntax: KSEA.HAROB5.ERAVE.Q1.EASON.Q1.ETCHY..MLBEC..BDEGA2.KSFO.

Flight plans for constructive and/or live traffic will be published to LVC Gateway by the HLA via the LVC Gateway Toolbox component that is part of the HLA distributed environment. In addition, any constructive, virtual or live UAS entity connecting to the LVC Gateway will generate and publish a flight plan in the `MsgFlightPlan` format. The Gateway will transmit UAS flight plans to the LVC Gateway Toolbox and the HLA environment. The message field, `m_adrsProc`, should be set to the corresponding enum data type `adrs_proc_type` defined in the `adrs_interface.h` header file provide to the user. The `m_adrsProc` is set to `ADRS_PROC_MPI_CLIENT_MACS = 35` if targets are generated by MACS while `ADRS_PROC_MPI_CLIENT_VAST = 38` if targets are generated external to MACS, i.e. by a federate from the HLA distributed environment.

The `m_status` field as a minimum should be set to `ADRS_MPI_FS_LNAV` which corresponds to the bit field for lateral navigation management. CSD will not function nominally if `m_status` is set to a zero value.

#### 4.3.6 Aircraft Flight Trajectory Intent Structure

The Aircraft Flight Trajectory Intent structure is defined below. It is a composite of two structures: 1) the trajectory specification structure, and 2) the waypoint structure. Both structures are defined below.

The Trajectory Intent Structure represents the payload of an aircraft flight trajectory intent message.

**Table 9. Trajectory Intent Message**

Data Type	Data Field Name	Description	Notes
MpiTraj Spec	<code>m_spec</code>	Trajectory Specification Structure	These data fields combined create the Trajectory Intent Message. The <code>m_point</code> field is limited to 50 waypoints.
MpiTraj Point	<code>m_point[ eMPI_MAX_TRAJ_POINTS=50 ]</code>	Trajectory Waypoint Structure	

**Table 10. Trajectory Specification Structure (MpiTrajSpec Structure)**

<b>Data Type</b>	<b>Data Field Name</b>	<b>Units</b>	<b>Description</b>	<b>Notes</b>
char	m_acid[ eMPI_ID_LEN GTH=12 ]		Aircraft Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_adrsProc		Defines the MACS internal process that denotes where a message originated from.	Values may be accessed in Appendix B, with a value of 38 to indicate a VAST connection.
int	m_cid		Computer Identification	Unique numerical identifier for each aircraft.
int	m_numberOfPoints		Total number of waypoints	
int	m_numberOfHoriz Points		Number of waypoints laterally and longitudinally.	
float	m_climbSpeed	feet per minute (fpm)	The rate of climb or descent of an aircraft indicated in feet per minute.	
float	m_cruiseSpeed	knots	Airspeed of an aircraft in level flight after climb to a set altitude and before descent.	
float	m_descentSpeed	knots	Airspeed of an aircraft during descent.	
float	m_approachSpeed	knots	The speed of the aircraft when making an approach to landing.	
float	m_landingSpeed	knots	The speed of the aircraft after landing.	
float	m_cruiseAltitude	feet	The height at which an airplane stays for most of a flight.	
float	m_currentGross Weight		The total aircraft weight at any moment during the flight or ground operation.	Not supported
float	m_miscFloatValue			Not supported
char	m_text[ eMPI_STRING_ TRAJ=128 ]			Add descriptive text as needed

**Table 11. Trajectory Waypoint Structure (MpiTrajPoint Structure)**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
eMpiTrajPtType	m_type				Defined by 4 byte enumeration that defines waypoint types in the MpiTrajPoint structure.	See the eMpiTrajPtType table for definitions.
char	m_waypointId[eMPI_ID_LENGTH=12]				Name of the fix, waypoint, or FRD	
float	m_latitude	Decimal degrees	-90 <sup>o</sup>	+90 <sup>o</sup>	Latitude of waypoint	Northern latitudes are positive, Southern latitudes are negative  Expressed to 6 decimal places. Example: 43.068286
float	m_longitude	Decimal degrees	-180 <sup>o</sup>	+180 <sup>o</sup>	Longitude of waypoint	Eastern longitudes are positive, Western longitudes are negative  Expressed to 6 decimal places. Example: 43.068286
float	m_turnRadius	Degrees	0	360	The horizontal distance that an aircraft uses to turn is referred to as the radius of turn.	Not supported
int	m_miscIntValue					
double	m_eta	UTC seconds			Estimated Time of Arrival at the waypoint.	
float	m_calibratedAirSpeed	Knots			Indicated airspeed corrected for instrument and position error.	
float	m_altitude	Feet			The height of the aircraft in relation to the mean sea level (MSL).	
float	m_fuelRemaining	Pounds			The amount of fuel remaining on board an aircraft	Not supported
float	m_outerAirTemperature	Fahrenheit			The temperature of the air around an aircraft	Not supported
float	m_windDirection	Degrees	0	360	Reported by the direction from which it originates in azimuth degrees	To be determined
float	m_windSpeed	Knots			Air moving from high pressure to low pressure	To be determined
float	m_trueAirSpeed	Knots			The speed of the aircraft relative to the air mass in which it is flying.	

Data Type	Data Field Name	Units	Min	Max	Description	Notes
float	m_trueCourseIntoPoint	Degrees			The course of the aircraft to the waypoint measured with respect to true north.	Not supported by MACS. Used for “heading” Trial Planner. Set to -999999.0 when not used otherwise, set to the trial angle.
float	m_distanceToPoint	NM			The distance of the aircraft to the waypoint	TBD: subject to computation. MACS uses EntryTime.
float	m_predictedGrossWeight	Pounds			The predicted total aircraft weight at the time the aircraft crosses the waypoint.	Not supported by MACS.
float	m_x					To be determined
float	m_y					To be determined
int	m_constraint					
float	m_miscFloatValue					

The **MpiTrajectory** of the constructive and/or live traffic is published by the HLA via the LVC Gateway Portal component to the LVC Gateway. The Gateway will publish the trajectory intent of any constructive, virtual or live UAS entity connecting to the LVC Gateway. Subsequently, **MsgTrajectory** messages associated with UAS entities will be transmitted to the LVC Gateway Portal and HLA environment. The message field, **m\_adrsProc**, should be set to the corresponding data source value defined in Table B1 in Appendix B.

*Note: MACS does not follow the conventions for latitude and longitude specified in the structures defined above. The expected format by MACS is Degrees:Minutes:Seconds. The LVC Gateway accounts for this variance in messages to MACS via ADRS.*

Enumeration below defines waypoint types in the **MpiTrajPoint** structure. The size of the enumeration field is 4 bytes.

**Table 12. Trajectory Waypoint Types**

Data Type	Value	Data Field Name	Description	Notes
int	0	eMPI_TRAJ_TYPE_WP	Defines waypoint type at a waypoint.	
int	1	eMPI_TRAJ_TYPE_HP	Defines waypoint type at a holding pattern.	A racetrack pattern for delay of aircraft in flight.
int	2	eMPI_TRAJ_TYPE_PH	Defines waypoint type at a procedure hold.	A prescribed hold at a designated fix.
int	3	eMPI_TRAJ_TYPE_PT	Defines waypoint type at a procedure turn.	The maneuver prescribed when it is necessary to perform a course reversal to establish the aircraft inbound on an intermediate or final approach course.
int	4	eMPI_TRAJ_TYPE_RF	Defines waypoint type at an RF leg.	Constant radius turns around a fix are called "radius-to-fix legs," or RF legs.
int	5	eMPI_TRAJ_TYPE_TC	Defines waypoint type at the ATC.	Point where aircraft is predicted to level off at the assigned altitude.
int	6	eMPI_TRAJ_TYPE_TD	Defines waypoint type at ATD.	Point where an aircraft is expected to begin descent from its assigned altitude.
int	7	eMPI_TRAJ_TYPE_SL	Defines waypoint type at the start of level flight.	
int	8	eMPI_TRAJ_TYPE_CA	Defines waypoint type at the crossover altitude.	The altitude where a specific Indicated Airspeed (IAS) becomes a specific Mach number (M).
int	9	eMPI_TRAJ_TYPE_TA	Defines waypoint type at the transition altitude.	The altitude at or below which the vertical position of an aircraft is controlled by reference to altitudes.
int	10	eMPI_TRAJ_TYPE_AC	Defines waypoint type at the aircraft's position.	
int	11	eMPI_TRAJ_TYPE_CS	Defines waypoint type at a constraint.	
int	12	eMPI_TRAJ_TYPE_RT	Defines waypoint type at a part of the current route.	
int	13	eMPI_TRAJ_TYPE_AP	Defines waypoint type at airport data.	
int	14	eMPI_TRAJ_TYPE_SC	Defines waypoint type at a speed change point.	

This data type is defined in the `adrs_trajectory.h` interface file provided to the user.

#### 4.3.7 Aircraft Delete Structure

The Aircraft Delete structure is defined below. This structure represents the payload of an aircraft delete message.

**Table 13. Delete Aircraft Message**

Data Type	Data Field Name	Description	Notes
char	m_acid[eMPI_ID_LENGTH=12]	Aircraft Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_adrsProc	Defines the MACS internal process that denotes where a message originated from.	Values may be accessed in Appendix B, with a value of 38 to indicate a VAST connection.
int	m_cid	Computer Identification	Unique numerical identifier for each aircraft.

An aircraft may drop out of the simulation environment due to a process crash, operational reasons (intentional shut down of the process) or during the debugging process. The LVC Gateway will send the MsgDeleteAc message to all clients that subscribe to the delete message. This message is initiated by the LVC Gateway clients: Ikhana GCS, Ikhana Sim, Langley UAS, or VIRTUAL UAS and sent to the LVC Gateway. These processes can reconnect and continue participating in the same ongoing simulation.

In addition, the Delete message can be initiated by the HLA distributed environment when the aircraft from the background traffic drops out of the simulation. This event will generate a Delete message in the HLA environment which will be propagated throughout the entire distributed system. This informs the system components that the HLA aircraft is no longer active and that the local instance of the aircraft should be removed.

In case of the lost link between the Ikhana GCS and the Ikhana aircraft the Delete message will be sent from the IOServer/CPDS to the Gateway. The IOServer/CPDS is an interface between the LVC and Live domains during flight tests.

#### 4.3.8 Set Ownership Structure

The Aircraft Ownership structure is defined below. This structure represents the payload of an aircraft set-ownership message.

**Table 14. Set Ownership Message**

Data Type	Data Field Name	Description	Notes
char	m_acid[eMPI_ID_LENGTH=12]	Aircraft Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
char	m_host[eMPI_STRING_HOST=24]	The name of the system that provides the ownership to the gateway.	
int	m_cid	Computer Identification	Unique numerical identifier for each aircraft.
int	m_control		

This message is deprecated as CSD (CDTI) system component is not used for tests any more. It is maintained in the ICD for back compatibility in case if CSD is reused in the LVC environment in future. The message is used to inform a CDTI system component about the target it is associated with. The CDTI will initially provide the ownship call sign by the handshake message sent to the LVC Gateway. The ownship call sign is specified by the dataProviderName data field. Upon receiving handshake message, the Gateway will generate MsgSetOwnship message using the received call sign and the CID corresponding to the target with the specified call sign and will send it back to CDTI.

#### 4.3.9 Sense and Avoid (SAA) Threat Results Message

The SAA Threat Results Message data structure is defined below. It is a composite of two structures: 1) the threat specification data structure, and 2) the threat data structure. Both structures are defined below.

The SAA Threat Results Message Structure represents the payload comprised of array of SaaThreat data structures defined below.

**Table 15. SAA Threat Results Message**

Data Type	Data Field Name	Description	Notes
Saa ThreatSpec	m_spec	Threat Specification Structure	
Saa Threat	m_threats[SAA_MAX_THREATS=50]	Threat Data Structure	SAA Max Threats is arbitrary, change as required

**Table 16. SAA Threat Specification Structure**

Data Type	Data Field Name	Description	Notes
char	m_acid[eMPI_ID_LENGTH=12]	Aircraft Call Sign, Ownship	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_cid	Computer Identification, Ownship	Unique numerical identifier for each aircraft.
int	m_numberOfThreats	Total number of threats included in the message	Maximum number of threats limited to 50. May be adjusted, as required.

*Note that the eSaaType in the table below is a type defined as an int, i.e. typedef int eSaaType.*

Table 17. SAA Threat Message

Data Type	Data Field Name	Units	Min	Max	Description	Notes
eSaa Type	m_saaType				SAA alert level	Type defined as an int.
int	m_intruderCid				Computer Identification, Intruder	
double	m_conflictStartTime	UTC seconds			Conflict start time	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
double	m_conflictEndTime	UTC seconds			Conflict End Time	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
double	m_conflictDuration	Seconds			Total Length of the conflict	
double	m_timeToCpa	Seconds			Time to reach the Closest Point of Approach	
double	m_timeToFirstLoss	Seconds			Time to violation of the Well Clear envelope.	Loss of Well Clear occurs 35 seconds from intruder within 450 feet.
double	m_dTauSimple	Seconds			Time to CPA between the ownship and the intruder.	Range divided by range rate
double	m_dTauModified	Seconds			Tau based on horizontal ranges not slant ranges	Range divided by range rate
float	m_minHorizSep	NM			Minimum distance laterally and longitudinally required to remain well clear of an intruder	
float	m_minVertSep	Feet			Minimum distance vertically required to remain well clear of an intruder	
double	m_ownshipCpaLat	Decimal degrees	-90 <sup>0</sup>	+90 <sup>0</sup>	Latitude of the ownship at the closest point of approach.	Northern latitudes are positive, Southern latitudes are negative  Expressed to 6 decimal places. Example: 43.068286
double	m_ownshipCpaLon	Decimal degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Longitude of the ownship at the closest point of approach.	Eastern longitudes are positive, Western longitudes are negative.  Expressed to 6 decimal places. Example: 43.068286
double	m_intruderCpaLat	Decimal degrees	-90 <sup>0</sup>	+90 <sup>0</sup>	Latitude of the intruder at the closest point of approach.	Northern latitudes are positive, Southern latitudes are negative.  Expressed to 6 decimal places. Example: 43.068286

Data Type	Data Field Name	Units	Min	Max	Description	Notes
double	m_intruderCpaLon	Decimal degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Longitude of the intruder at the closest point of approach.	Eastern longitudes are positive, Western longitudes are negative.  Expressed to 6 decimal places. Example: 43.068286
double	m_ownershipFristLossLat	Decimal degrees	-90 <sup>0</sup>	+90 <sup>0</sup>	Latitude of the ownship at the point where the Well Clear envelope is violated.	Northern latitudes are positive, Southern latitudes are negative.  Expressed to 6 decimal places. Example: 43.068286
double	m_ownershipFristLossLon	Decimal degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Longitude of the ownship at the point where the Well Clear envelope is violated.	Eastern longitudes are positive, Western longitudes are negative.  Expressed to 6 decimal places. Example: 43.068286
double	m_intruderFristLossLat	Decimal degrees	-90 <sup>0</sup>	+90 <sup>0</sup>	Latitude of the intruder at the point where the Well Clear envelope is violated.	Northern latitudes are positive, Southern latitudes are negative.  Expressed to 6 decimal places. Example: 43.068286
double	m_intruderFristLossLon	Decimal degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Longitude of the intruder at the point where the Well Clear envelope is violated.	Eastern longitudes are positive, Western longitudes are negative.  Expressed to 6 decimal places. Example: 43.068286
float	m_ownershipCpaAlt	Feet			Altitude of the ownship at the CPA.	
float	m_ownershipCpaGroundSpeed	Knots			The ownship's airspeed relative to the ground at CPA.	
float	m_ownershipCpaCalibratedAirSpeed	Knots			The ownship's indicated airspeed corrected for instrument and position error at CPA.	
float	m_ownershipCpaVerticalSpeed	Feet per minute (fpm)			The rate of climb or descent of the ownship at the CPA.	
float	m_ownershipCpaHeading	Degrees	0	360	The ownship's heading at CPA.	
float	m_intruderCpaAlt	Feet			Altitude of the intruder at the closest point of approach	
float	m_intruderCpaGroundSpeed	Knots			The intruder's airspeed relative to the ground at CPA.	
float	m_intruderCpaCalibratedAirSpeed	Knots			The intruder's indicated airspeed corrected for instrument and position error at CPA.	

Data Type	Data Field Name	Units	Min	Max	Description	Notes
float	m_intruderCpa VerticalSpeed	Feet per minute (fpm)			The rate of climb or descent of the intruder at the CPA.	
float	m_intruderCpa Heading	Degrees	0	360	The intruder's heading at CPA.	
float	m_ownshipFirst LossAlt	Feet			Altitude of the ownship at the point where the Well Clear envelope is violated.	
float	m_ownshipFirstLoss GroundSpeed	Knots			The ownship's airspeed relative to the ground at the point where the Well Clear envelope is violated.	
float	m_ownshipFirstLoss CalibratedAirSpeed	Knots			The ownship's indicated airspeed corrected for instrument and position error at the point where the Well Clear envelope is violated.	
float	m_ownshipFirstLoss VerticalSpeed	Feet per minute (fpm)			The rate of climb or descent of the ownship at the point where the Well Clear envelope is violated.	
float	m_ownshipFirstLoss Heading	Degrees	0	360	The ownship's heading at the point where the Well Clear envelope is violated.	
float	m_intruderFirstLoss Alt	Feet			Altitude of the intruder at the point where the Well Clear envelope is violated.	
float	m_intruderFirstLoss GroundSpeed	Knots			The intruder's airspeed relative to the ground at the point where the Well Clear envelope is violated.	
float	m_intruderFirstLoss CalibratedAirSpeed	Knots			The intruder's indicated airspeed corrected for instrument and position error at the point where the Well Clear envelope is violated.	
float	m_intruderFirstLoss VerticalSpeed	Feet per minute (fpm)			The rate of climb or descent of the intruder at the point where the Well Clear envelope is violated.	
float	m_intruderFirstLoss Heading	Degrees	0	360	The intruder's heading at the point where the Well Clear envelope is violated.	
bool	m_isPredictedStricter					
char	pad[7]					

The definition of DAA alert levels are defined in the table below.

**Table 18. Alert Levels, Definitions**

Data Type	Alert Level	Description
int	0	No alert
int	1	Proximate alert
int	2	Preventative Self-separation alert
int	3	Corrective Self-separation alert
int	4	Warning Self-separation alert

The color scheme, symbology, for different alert levels is presented in Appendix A.

#### 4.3.10 SAA Release Structure

When the SAA algorithm determines that SS or CA threat no longer exists, the Sense and Avoid algorithm returns RELEASE as the threat state for the ownship. SaaProc generates the SAA Release message, indicating that the conflict has been cleared and sends the MPI Release Message to the LVC Gateway. The threat symbology is subsequently removed from the CSD or VSCS displays.

*Note that the eSaaType in the table below is a type defined as an int, i.e. typedef int eSaaType.*

**Table 19. SAA Release Message**

Data Type	Data Field Name	Description	Notes
char	m_acid[eMPI_ID_LENGTH=12]	Aircraft Callsign, Ownship	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_cid	Computer Identification, Ownship	Unique numerical identifier for each aircraft.
eSaaType	m_saaType	SAA alert level	Type defined as an int.

#### 4.3.11 Sense and Avoid (SAA) Resolution Maneuver Structure

The SAA Resolution Maneuver data structure is defined below. It is a composite of two structures: 1) the maneuver specification data structure, and 2) the maneuver data structure. Both structures are defined below.

The **MsgSaaResManeuver** structure describes the payload of a SAA Resolution Maneuver message.

**Table 20. SAA Resolution Maneuver Message**

Data Type	Data Field Name	Description	Notes
SaaResManeuverSpec	m_maneuverSpec	Maneuver Specification Structure	
SaaManeuver	m_maneuvers[SAA_MAX_MANEUVERS=20]	Maneuver Data Structure	Maximum number of maneuvers is 20

**Table 21. SAA Resolution Maneuver Specification**

Data Type	Data Field Name	Description	Notes
char	m_acid[eMPI_ID_LENGTH=12]	Aircraft Call Sign, Ownship	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_ownershipCid	Computer Identification, Ownship	Unique numerical identifier for each aircraft.
int	m_numberOfManeuvers	Total number of aircraft maneuvers	

**Table 22. SAA Maneuver**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
eManeuver Type	m_maneuverType				Maneuver type	As enumerated in eManeuverType
eSaaType	m_saaType				SAA alert level	Type defined as an int.
double	m_startTime	UTC seconds			Start time of the SAA maneuver	
double	m_endTime	UTC seconds			End time of the SAA maneuver	
float	m_altitude	Feet			Altitude of the maneuver	Not set (-999999) if maneuver type != altitude
float	m_headingAbs	Degrees	0	359	Current true heading of the ownship	Not set (-999999) if maneuver type != heading
float	m_headingRel	Degrees	0	±180	The clockwise or counterclockwise angle of the resolution maneuver in degrees from the current true heading of the ownship.	Right turn expressed as positive; Left turn expressed as a negative. Not set (-999999) if maneuver type != heading
float	m_speed	Knots			Speed of the maneuver	Not set if maneuver type != speed

**Table 23. SAA Maneuver Type**

Data Type	Enum Value	Data Field Name	Description	Notes
int	0	eSAA_MANEUVER_TYPE_REROUTE	Reroute maneuver	
int	1	eSAA_MANEUVER_TYPE_HEADING	Maneuver turns to a new heading	
int	2	eSAA_MANEUVER_TYPE_SPEED	Maneuver commands a new airspeed	
int	3	eSAA_MANEUVER_TYPE_ALTITUDE	Maneuver changes the altitude of the aircraft	
int	4	eSAA_MANEUVER_TYPE_COMPOUND	Maneuver consists of any combination of changes to route, heading, altitude, or speed.	
int	5	eSAA_MANEUVER_TYPE_APP_REFINED		
int	-999999	eSAA_MANEUVER_TYPE_NOT_SET	No maneuver	Same as INT_NOT_SET



#### 4.3.12 Sense and Avoid (SAA) Resolution Reroute Structure

The SAA Resolution Reroute data structure is defined below. It is a composite of two structures: 1) the resolution reroute specification data structure, and 2) the resolution waypoints data structure. Both structures are defined below. The **MsgSaaResReroute** structure represents the payload of an aircraft reroute message from a DAA resolution.

**Table 24. SAA Resolution Reroute Message**

Data Type	Data Field Name	Description	Notes
SaaResRerouteSpec	m_rerouteSpec	The resolution reroute specification data structure	
SaaResWaypoint	m_waypoints[MPI_MAX_NUMBER_OF_WAYPOINTS=50]	The resolution waypoints data structure	Maximum number of waypoints is 50

**Table 25. SAA Resolution Reroute Specifications**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
char	m_acid[eMPI_ID_LENGTH=12]				Aircraft Call Sign, Ownship	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_cid				Computer Identification, Ownship	Unique numerical identifier for each aircraft.
eSaaType	m_saaType				SAA alert level	Type defined as an int.
double	m_startTime	UTC seconds			Start time of the SAA maneuver	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
double	m_endTime	UTC seconds			End time of the SAA maneuver	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
int	m_numberOfWaypoints				Total number of waypoints	
double	m_turnOutAngle	Degrees	0	±180	Turn angle to the next fix from the current location	Right turn expressed as positive; Left turn expressed as a negative.

**Table 26. SAA Resolution Waypoint Structure**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
char	m_name[eMPI_ID_LENGTH=16]				Waypoint Name	May use the actual name of the waypoint or assign one arbitrarily if none available.
double	m_latitude	Decimal degrees	-90 <sup>0</sup>	+90 <sup>0</sup>	Latitude of ownship	Northern latitudes are positive, Southern latitudes are negative. Expressed to 6 decimal places. Example: 43.068286
double	m_longitude	Decimal degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Longitude of ownship	Eastern longitudes are positive, Western longitudes are negative. Expressed to 6 decimal places. Example: 43.068286
float	m_altitude	Feet			The altitude of the ownship.	Express as above Mean Sea Level (MSL)
float	m_speed	Knots			True airspeed of the ownship	

#### 4.3.13 Navigation Mode Message Structure

The Navigation Mode Message is used whenever the ownship flight control system executes a maneuver or when the ownship consumes a waypoint on the route. The purpose is to send the SAA system intent information, so it can build an accurate trajectory prediction while detecting threats.

Table 27. Navigation Mode Message

Data Type	Data Field Name	Mode				Units	Min	Max	Description	Notes
		Flightplan	Manual	Autopilot	Override					
eNav Mode	m_eNavMode	X	X	X	X				Navigation Mode	Enumerated in accordance with the eNavMode Table. Mandatory field for all modes.
char	m_acid[eMPI_ID_LENGTH=12]	X	X	X	X				Aircraft Call Sign, Ownship	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61 Mandatory field for all modes.
int	m_ownershipCid	X	X	X	X				Computer Identification, Ownship	Unique numerical identifier for each aircraft. Mandatory field for all modes.
float	m_heading			X		Degrees	0	359	Ownship True Heading	If not in Autopilot Mode, set to -999999
float	m_altitude			X		Feet			The altitude of the ownship.	If not in Autopilot Mode, set to -999999. Expressed as above Mean Sea Level (MSL)
float	m_override Altitude				X	Feet			The revised altitude of the ownship, when overridden.	If not in Override Mode, set to -999999. Expressed as above Mean Sea Level (MSL)
float	m_tas				X	Knots			Speed of the ownship relative to the air mass in which it is flying.	If not in Override Mode, set to -999999. At least one speed must be set.
float	m_cas				X	Knots			Indicated airspeed of the ownship corrected for instrument and position error.	
float	m_mach				X	Knots			Ownship's airspeed relative to the speed of sound	

**Table 28. Navigation Mode**

Data Type	Data Field Name	Value	Description	Notes
int	eNAV_MODE_FLIGHT_PLAN	0	Aircraft navigates using a route defined by a series of waypoints.	
int	eNAV_MODE_AUTO_PILOT	1	Aircraft navigates using VNAV and LNAV.	
int	eNAV_MODE_OVERRIDE	2	Altitude and/or speed setting are overwritten at the GCS.	GCS provides updated waypoint lists followed by override msg.
int	eNAV_MODE_MANUAL	3	Aircraft navigates using manual inputs by the pilot.	
int	eNAV_MODE_NOT_SET	-999999	No Nav Mode setting	same as INT_NOT_SET

#### 4.3.14 ACAS\_Xu Data Structures

The Acasxu wrapper provides updates for aircraft track state messages for the ownship and intruders while the ACAS-Xu algorithm generates the MsgAcasxu output message defined below. The message provides Vertical Resolution Advisory, Horizontal Resolution Advisory, Preventive Horizontal Collision Avoidance Band, and Preventive Horizontal Remain-Well-Clear Band for the given traffic update.

The value for ACASXU\_MAX\_INTRUDERS is set to 64.

**Table 29. ACAS-Xu Alerting and Guidance Message**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
<b>Vertical Resolution Advisory provided by ARINC Label 270</b>						
char	m_ownershipCallsign[MAX_CALLSIG_LEN=12]				Ownship callsign	
double	m_timeOfApplicability	Decimal seconds			Time at which the alert message was applied.	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places. Example: 474041713.363
double	m_timeReceived	Decimal seconds			Time when the Flight State Message was received.	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.

Data Type	Data Field Name	Units	Min	Max	Description	Notes
TcasAural Alert	m_auralAlertLevel				TCAS aural annunciation	
int	m_escapeVspd	Feet per minute (fpm)	-6,300	6,300	TCAS vertical escape rate to maintain safe separation.	Function of aircraft type and performance
int	m_combinedControl				Indicates the type of Resolution Advisory (RA)	See TCAS II 7.1 combined control
int	m_verticalControl				Indicates modifications to either increase/decrease the strength or reverse the sense of the initial RA.	See TCAS II 7.1 vertical control
int	m_downAdvisory				Defines value for downward sense alerts.	See TCAS II 7.1 down advisory flag
int	m_upAdvisory				Defines value for upward sense alerts.	See TCAS II 7.1 up advisory flag
int	m_numberOfIntruders				Total number of intruders	
int	m_intruderUID[ MAX_NUM_OF_ INTRUDERS]				List of intruder data.	Max array size is MAX_NUMBER_OF_INTRUDERS
<b>Horizontal Resolution Advisory provided by ARINC Label 271</b>						
int	m_advisoryHeading	Degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Target heading relative to true North	The goal heading (clockwise of true North is positive). The provided heading angle shall be adjusted to magnetic declination and rounded to the nearest 5 degree multiple by the application driving the display.
int	m_enumCombinedControl		0	7	Enumerates the type of Resolution Advisory (RA)	See the Horizontal Resolution Advisory Combined Control table below.
int	m_enumHorizontal_RA_AuralAlert (optional)		0	6	Enumerates the Aural Alert for the horizontal RA	See the Horizontal Resolution Advisory Aural Alert table below.

Data Type	Data Field Name	Units	Min	Max	Description	Notes
<b>Preventive Horizontal Collision Avoidance Band Information provided by ARINC Label 275</b>						
bool	m_preventive CABands[20]	Degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Each indicated bit set to 1 in the list of heading intervals will be illuminated in RED measured from the current ownship heading. Each bit value (0 or 1) is represented by 10 Degrees.	
<b>Preventive Horizontal Remain-Well-Clear Band Information provided by ARINC Label 276</b>						
bool	m_preventive RWCbands[20]	Relative Degrees	-180 <sup>0</sup>	+180 <sup>0</sup>	Each indicated bit set to 1 in the list of band intervals will be illuminated in YELLOW measured from the current ownship heading. Each bit value (0 or 1) is represented by 10 Degrees.	

**Table 30. Vertical Resolution Advisory - Combined Control**

Data Type	Enum Value	Data Field Name	Description
int	0	COMB_CONT_NONE	No Advisory
int	1	CLEAR_OF_CONFLICT	Traffic conflict is resolved and RA is removed.
int	2	SPARE	Not Used
int	3	SPARE	Not Used
int	4	CLIMB_CORR	RA requiring positive change in vertical speed
int	5	DESCEND_CORR	RA requiring negative change in vertical speed
int	6	PREVENTATIVE	RA requiring no change in vertical speed

**Table 31. Vertical Resolution Advisory - Vertical Control**

Data Type	Enum Value	Data Field Name	Description
int	0	VERT_CONT_NONE	None of the following vertical controls
int	1	CROSSING	Indicates that the ownship is crossing through the intruder's altitude.
int	2	REVERSE	Encounter in which it is necessary to reverse the original RA to avoid the threat
int	3	INCREASE	Increase in the strength of the initial RA to avoid a threat
int	4	MAINTAIN	RA requiring no change in vertical speed to avoid a threat

**Table 32. Vertical Resolution Advisory - Up Advisory**

Data Type	Enum Value	Data Field Name	Description	Notes
int	0	UP_ADV_NONE	No up advisory	
int	1	CLIMB	Climb @ 1500 to 2000 fpm	
int	2	DONT_DESCEND	Do not descend	
int	3	DONT_DESCEND500	Do not descend faster than 500 fpm	*Not used for ACAS-Xu
int	4	DONT_DESCEND1000	Do not descend faster than 1000 fpm	
int	5	DONT_DESCEND2000	Do not descend faster than 2000 fpm	*Not used for ACAS-Xu

**Table 33. Vertical Resolution Advisory - Down Advisory**

Data Type	Enum Value	Data Field Name	Description	Notes
int	0	DOWN_ADV_NONE	No down advisory	
int	1	DESCEND	Descend @ 1500 to 2000 fpm	
int	2	DONT_CLIMB	Do not climb	
int	3	DONT_CLIMB500	Do not climb faster than 500 fpm	*Not used for ACAS-Xu
int	4	DONT_CLIMB1000	Do not climb faster than 1000 fpm	
int	5	DONT_CLIMB2000	Do not climb faster than 2000 fpm	*Not used for ACAS-Xu

**Table 34. Horizontal Resolution Advisory - Combined Control**

Data Type	Enum Value	Data Field Name	Description	Notes
int	0	No Advisory	No Advisory	
int	1	Clear of Conflict	Traffic conflict is resolved and RA is removed.	
int	2	SPARE	Not Used	
int	3	SPARE	Not Used	
int	4	Right Collision Avoidance Advisory	RA requiring positive change in heading	Example: +30 <sup>0</sup>
int	5	Left Collision Avoidance Advisory	RA requiring negative change in heading	Example: -30 <sup>0</sup>
int	6	Right Self Separation Advisory	WCR requiring positive change in heading	Example: +30 <sup>0</sup>
int	7	Left Self Separation Advisory	WCR requiring negative change in heading	Example: -30 <sup>0</sup>

**Table 35. Horizontal Resolution Advisory Aural Alerts**

<b>Data Type</b>	<b>Enum Value</b>	<b>RA Type</b>	<b>Aural Annunciation</b>	<b>Description</b>	<b>Visual Indication</b>
int	0		No Advisory	No advisory	Visual Indication: No Advisory
int	1		Clear of Conflict	Traffic conflict is resolved and RA is removed.	Clear of Conflict
int	4	Corrective CA	Turn Right, Turn Right	RA requiring positive change in heading	Turn Right, Heading (###)(Red Text)
int	5	Corrective CA	Turn Left, Turn Left	RA requiring negative change in heading	Turn Left, Heading (###)(Red Text)
int	6	Corrective RWC	Turn Right, Turn Right	RWC requiring positive change in heading	Turn Right, Heading (###)(Yellow Text)
int	7	Corrective RWC	Turn Left, Turn Left	RWC requiring negative change in heading	Turn Left, Heading (###)(Yellow Text)

Table 36. Vertical Resolution Advisory Interpretation

Data Type	Enum Value	Advisory	RA Type	Aural Annunciation	Visual Indication	Rate to Maintain (fpm)	Combined Control	Vertical Control	Up Advisory	Down Advisory
int	0	Clear of Conflict		Clear of Conflict	CLEAR OF CONFLICT		1	0	0	0
int	1	Climb	Corrective	Climb, Climb	CLIMB, 1000 fpm	+1000	4	0	1	0
int	2	Reduce Descent (Do Not Descend)	Corrective	Level Off, Level Off	LEVEL OFF	0	4	0	2	0
int	3	Altitude Crossing Climb	Corrective	Climb, Crossing Climb; Climb, Crossing Climb	CROSSING CLIMB 1000 fpm	+1000	4	1	1	0
int	4	RA Reversal (Descend to Climb)	Corrective	Climb, Climb NOW; Climb, Climb NOW	CLIMB NOW 1000 fpm	+1000	4	2	1	0
int	5	Descend	Corrective	Descend, Descend	DESCEND, -1000 fpm	-1000	5	0	0	1
int	6	Reduce Climb (Do Not Climb)	Corrective	Level Off, Level Off	LEVEL OFF	0	5	0	0	2
int	7	Altitude Crossing Descend	Corrective	Descend, Crossing Descend; Descend, Crossing Descend	CROSSING DESCEND, -1000 fpm	-1000	5	1	0	1
int	8	RA Reversal (Climb to Descend)	Corrective	Descend, Descend NOW; Descend, Descend NOW	DESCEND NOW, -1000 fpm	-1000	5	2	0	1
int	9	Limit Climb (Do Not Climb)	Preventive	Monitor Vertical Speed	MONITOR V/S		6	0	0	2
int	10	Limit Descend (Do Not Descend)	Preventive	Monitor Vertical Speed	MONITOR V/S		6	0	2	0
int	11	Multi-Aircraft Encounter (Issued While Climbing)	Corrective	Level Off, Level Off	LEVEL OFF	0	5	0	2	2
int	12	Multi-Aircraft Encounter (Issued While Descending)	Corrective	Level Off, Level Off	LEVEL OFF	0	4	0	2	2
int	13	Multi-Aircraft Encounter	Preventive	Monitor Vertical Speed	MONITOR V/S		6	0	2	2
int	14	Weakening of Positive RA (After Up Sense RA)	Corrective	Level Off, Level Off	LEVEL OFF	0	4	0	2	0
int	15	Weakening of Positive RA (After Down Sense RA)	Corrective	Level Off, Level Off	LEVEL OFF	0	5	0	0	2

**Table 37. TCAS Alert Levels**

Data Type	Enum Value	Data Field Name	Description	Notes
int	0	TCAS_THREAT_NONE	Other Traffic, altitude unknown.	
int	1	TCAS_THREAT_PROX	Proximate Traffic - non-threat traffic within 6 nm and ±1200 feet from the ownship.	As in TCAS 7.0
int	2	TCAS_THREAT_TA	Intruders that cause an TA to be issued (Mode A only)	As in TCAS 7.0
int	3	TCAS_THREAT_TA_MODE_C	Intruders that cause an TA to be issued (Mode A & C)	As in TCAS 7.0
int	4	TCAS_THREAT_RA	Intruders that cause an RA to be issued	As in TCAS 7.0

#### 4.3.15 Stratway Bands Data Structure

New terminology for the Stratway subsystem has been introduced. Stratway has been renamed to DAIDALUS. However, it has been decided to retain all the legacy references to Stratway in all of the pertinent messages in this ICD. This preserves and maintains consistent terminology between the current ICD and the software that has been developed using the previous version of the ICD for the earlier phases of the UAS-in-the-NAS project.

The original Stratway Bands message is defined in the Stratway+ External Interface (Stratway+ ExternalInterface\_Dec\_22) ICD provided by LaRC team.

The LVC Gateway will receive the Stratway Bands message from the Stratway+ GCS. A UDP client/server multicast protocol is used to send/receive Stratway+ bands data. In this configuration, Stratway+ GCS socket is a server while LVC Gateway socket is a client.

LVC Gateway will transmit the Stratway+ bands data to the subscribing clients based on the following Stratway+ Bands Message definitions.

The STRWAY\_MAX\_INTERVALS and STRWAY\_MAX\_INTRUDERS are set to a value of 10 imposed by the Stratway requirements.

*The character array in the table is defines as an array of 8 characters, char CharString8Type[8].*

**Table 38. Stratway Interval Type**

Data Type	Enum Value	Data Field Name	Description
int	0	eSTRWAY_INTERVAL_TYPE_UNKNOWN	Interval type not recognized
int	1	eSTRWAY_INTERVAL_TYPE_NONE	No Interval Type
int	2	eSTRWAY_INTERVAL_TYPE_NEAR	Near term conflict interval
int	3	eSTRWAY_INTERVAL_TYPE_RECOVERY	Recovery maneuver

**Table 39. Stratway Interval**

Data Type	Data Field Name	Units	Description	Notes
eIntervalType	m_eIntervalType		Indicates the color of the alert level to be displayed	
double	m_low_interval	Degrees or Feet above MSL	Indicates minimum value in degrees or feet relative to current ownship heading and altitude.	Example: -30 <sup>0</sup> or -1000 feet
double	m_up_interval	Degrees or Feet above MSL	Indicates maximum value in degrees or feet relative to current ownship heading and altitude.	Example: 30 <sup>0</sup> or 1000 feet

**Table 40. Stratway Intruder**

Data Type	Data Field Name	Description	Notes
CharString8Type	m_callsign	Aircraft Call Sign, Intruder	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
eSaaType	m_alertLevel	SAA alert level	Written as an int. Can have any value between 0 and 4 as defined in section 3.10.

*The stStratwayIntervalType is of typedef StratwayIntervalListType [STRWAY\_MAX\_INTERVALS] and the stStratwayIntruderType is of typedef StratwayIntruderListType [STRWAY\_MAX\_INTRUDERS]*

The Stratway+ bands data message that is sent from MACS's External Interface Communications Thread consists of the following data members:

**Table 41. Stratway Bands Message**

Data Type	Data Field Name	Units	Description	Notes
CharString8Type	m_callSign		Aircraft Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
double	m_timeSeconds	Seconds		
int	m_participantAddress			
int	m_numberOfHeadingIntervals		Total number of heading intervals	
StratwayIntervalListType	m_headingIntervalList		List of heading intervals	
int	m_numberOfTrueAirspeedIntervals		Total number of true airspeed intervals	
int	m_pad1			Padding used for data field alignment

Data Type	Data Field Name	Units	Description	Notes
Stratway Interval ListType	m_trueAirspeed IntervalList		List of true airspeed intervals	
int	m_numberOfVertical SpeedIntervals		Total number of Vertical Speed intervals	
int	m_pad2			
Stratway Interval ListType	m_verticalSpeed IntervalList		List of vertical speed intervals	
int	m_numberOfAltitude Intervals		Total number of Altitude intervals	
int	m_pad3			Padding for data field alignment
Stratway Interval ListType	m_altitudeIntervalList		List of altitude intervals	
int	m_numberOfIntruders		Total number of intruders	
int	m_pad4			
Stratway Interval ListType	m_intrudersList		List of intruders	

The data structure presented above is applicable for both 32 bit and 64 bit applications since alignment is 8-byte double-word aligned. However, padding has to be introduced in the structure to enforce the alignment.

#### 4.3.16 Omni (SAA) Band Message

The Omni (SAA) Band Message defines an interval with the same alert levels throughout (i.e. PROX, SS, CA, or NONE as defined by eSaaType). The interval is defined as [min, max] inclusive.

The same structure (OmniBandInterval) will be used to represent heading and altitude bands in the OmniBand concept (see MsgSaaOmniBands). For heading OmniBandInterval, min and max indicate heading in degrees relative to current ownship heading, e.g. -30 is 30 degrees left of ownship's current heading, and +30 is 30 degrees right of ownship's current heading. For altitude bands, min and max values will always be set to the same value as altitude bands represent a single altitude level in feet above MSL.

A heading OmniBandInterval indicates that between min and max the band should be colored according to the associated alertLevel. For example, if min = -45, max = 0, and alertLevel=0, then the interval from 45 degrees left of ownship to its current heading should be painted green.

For altitude OmniBandInterval, min = max, so either can represent the altitude level to be shown in the altitude menu and the alertLevel describes the color of its outline in the menu. For example, if min and max=15000 and alertLevel=3, this means that the altitude menu will include 15000 feet entry, whose outline should be red meaning ownship would cause a loss of well-clear if it maneuvers to 15,000 ft.

The SAA\_MAX\_BAND\_INTERVALS is set to a maximum value of 65.

**Table 42. Omni Bands Interval**

Data Type	Data Field Name	Units	Description	Notes
eSaaType	m_alertLevel		SAA alert level	Type defined as an int.
int	m_min	Degrees or Feet above MSL	Indicates minimum value in degrees or feet relative to current ownship heading and altitude.	Example: -30 <sup>0</sup> or -1000 feet
int	m_max	Degrees or Feet above MSL	Indicates maximum value in degrees or feet relative to current ownship heading and altitude.	Example: 30 <sup>0</sup> or 1000 feet

**Table 43. Omni Bands Message**

Data Type	Data Field Name	Units	Description	Notes
char	m_callsign[eMPI_ID_LENGTH]		Aircraft Call Sign, Ownship	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61
int	m_pad			
double	m_timeCreated	UTC Decimal seconds	Time when the Flight State Message was created.	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
int	m_altSuppressFlag			
int	m_pad			
int	m_numberOfHeading Intervals		Total number of heading intervals	
int	m_numberOfAltitude Intervals		Total number of Altitude intervals	
OmniBandInterval	m_headingIntervals[SAA_MAX_BAND_INTERVALS]		Indicates how the heading band is be colored according to the associated alertLevel	SAA_MAX_BAND_INTERVALS = 65, may be adjusted.
OmniBandInterval	m_altitudeIntervals[SAA_MAX_BAND_INTERVALS]		Indicates how the altitude band is be colored according to the associated alertLevel	

#### 4.3.17 TCAS 7.0 Message Structures

The FAA TCAS II v7.0 software application is integrated into the LVC system. It will produce a TCAS message output given the intruder inputs in a prescribed format defined above in section 3.10. The pertinent data structures and input requirements are defined below in this section. The TCAS application is a process that generates TA and RA output advisories as part of the `teas_alert_str` data structure.

The `MAX_CALLSIGN_LEN` is set to the value of 8.

Table 44. TCAS 7.0 Alert Structure

Data Type	Data Field Name	Units	Min	Max	Description	Notes
char	callsign[MAX_CALLSIGN_LEN=8]				Intruder Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61. Maximum length = 8
char	ac_type[MAX_CALLSIGN_LEN=8]				Alphanumeric code designating aircraft type	Maximum length = 8
double	timeCreated	UTC Decimal seconds			Time when the Flight State Message was created.	UNIX Epoch time in seconds since January 1, 1970. Include milliseconds by specifying value with 3 decimal places.
double	lat	Decimal degrees	-90 <sup>o</sup>	+90 <sup>o</sup>	Latitude of aircraft	Northern latitudes are positive, Southern latitudes are negative. Expressed to 6 decimal places. Example: 43.068286
double	lon	Decimal degrees	-180 <sup>o</sup>	+180 <sup>o</sup>	Longitude of aircraft	Eastern longitudes are positive, Western longitudes are negative. Expressed to 6 decimal places. Example: 43.068286
enum_tcas_threat_t	threat_status				Indicates the level of threat posed by the intruder.	Enumerated by the tcas_threat_t table.
float	rng	Seconds			Distance to the intruder aircraft based on tau.	
float	brg	Degrees			The direction of the intruder relative to the current heading of the ownship.	Bearing left of the current heading of the ownship is negative; to the right of the own ship is positive
int	alt	Feet above MSL			The altitude of the intruder	
int	ralt	Feet			The altitude if the intruder above or below the current altitude of the ownship	Altitude below the ownship is negative; above the ownship is positive.
int	arrow				The arrow expressing trend information for an intruder moving vertically.	-1 = down arrow; 0 is none; 1 is up arrow
int	vspd	Feet per minute (fpm)			The rate of climb or descent of the intruder aircraft.	Climb is indicated by positive number. Descent is a negative number.
float	hdg	Degrees	0	359	The true heading of the intruder	

Data Type	Data Field Name	Units	Min	Max	Description	Notes
int	gspd	Knots			Ground speed of the intruder	
int	ias	Knots			Indicated airspeed of the intruder	
int	xpdr_mode				Reply of the Intruder's transponder	
int	collision_time	Seconds			Time remaining before collision	
float	rate	Feet per minute (fpm)			Rate of climb or descent to maintain during a resolution maneuver.	
comb_cont_enum_t	combined_control				Indicates the type of Resolution Advisory (RA)	Enumerated in the comb_cont_enum_t table.
vert_cont_enum_t	vertical_control				Indicates modifications to either increase/decrease the strength or reverse the sense of the initial RA.	Enumerated in the vert_cont_enum_t table.
up_adv_enum_t	up_advisory				Defines the RA and the Required Vertical Rate in fpm for upward sense alerts.	Enumerated in the up_adv_enum_t table.
down_adv_enum_t	down_advisory				Defines the RA and the Required Vertical Rate in fpm for downward sense alerts.	Enumerated in the down_adv_enum_t table.
int	crossing				Indicates if the conflict is crossing the projected flight path of the ownship.	1 = crossing 2 = not crossing
tcas_aural_action_t	aural_action				Defines voice alert issued for displayed traffic alerts and RAs.	Enumerated in the tcas_aural_action_t table.
int	spare					

**Table 45. TCAS 7.0 Threat Table**

Data Type	Enum Value	Data Field Name	Description
int	0	TCAS_THREAT_NONE	Other Traffic, altitude unknown.
int	1	TCAS_THREAT_PROX	Proximate Traffic - non-threat traffic within 6 nmi and $\pm 1200$ ft from the ownship.
int	2	TCAS_THREAT_TA	Intruders that cause an TA to be issued (Mode A only)
int	3	TCAS_THREAT_TA_MODE_C	Intruders that cause an TA to be issued (Mode A and C)
int	4	TCAS_THREAT_RA	Intruders that cause an RA to be issued

**Table 46. TCAS 7.0 Aural Action Table**

Data Type	Enum Value	Data Field Name	Description	Notes
int	0	aural_NONE	No advisory	
int	1	aural_TRAFFIC_TRAFFIC	Traffic Advisory	
int	2	aural_MONITOR_VERTICAL_SPEED	Preventive RA	No change in vertical speed required
int	3	aural_CLIMB_CLIMB	Climb RA	
int	4	aural_CLIMB_CROSSING_CLIMB_CLIMB_CROSSING_CLIMB	Altitude Crossing Climb RA	
int	5	aural_DESCEND_DESCEND	Descend RA	
int	6	aural_DESCEND_CROSSING_DESCEND_DESCEND_CROSSING_DESCEND	Altitude Crossing Descend RA	
int	7	aural_REDUCE_CLIMB_REDUCE_CLIMB	Reduce Climb RA	"Adjust Vertical Speed"
int	8	aural_REDUCE_DESCENT_REDUCE_DESCENT	Reduce Descent RA	
int	9	aural_INCREASE_CLIMB_INCREASE_CLIMB	Increase Climb RA	
int	10	aural_INCREASE_DESCENT_INCREASE_DESCENT	Increase Descent RA	
int	11	aural_CLIMB_CLIMB_NOW_CLIMB_CLIMB_NOW	RA Reversal to Climb RA	
int	12	aural_DESCEND_DESCEND_NOW_DESCEND_DESCEND_NOW	RA Reversal to Descent RA	
int	13	aural_CLEAR_OF_CONFLICT	RA Removed	
int	14	aural_TCAS_SYSTEM_TEST_FAIL	TCAS system test failure	
int	15	aural_TCAS_SYSTEM_TEST_OK	TCAS system test passed	

Data Type	Enum Value	Data Field Name	Description	Notes
int	16	aural_ADJUST_VERTICAL_SPEED_ADJUST	Weakening of RA	
int	17	aural_MAINTAIN_VERTICAL_SPEED_CROSSING_MAINTAIN	Altitude Crossing, Maintain Rate [of climb or descent] RA	
int	18	aural_MAINTAIN_VERTICAL_SPEED_MAINTAIN	Maintain Rate [of climb or descent] RA	

Table 47. TCAS 7.0 Combined Control

Data Type	Enum Value	Data Field Name	Description
int	0	COMB_CONT_NONE	No Advisory
int	1	CLEAR_OF_CONFLICT	Traffic conflict is resolved and RA is removed.
int	2	CLIMB_CORR	RA requiring positive change in vertical speed
int	3	DESCEND_CORR	RA requiring negative change in vertical speed
int	4	PREVENTATIVE	RA requiring no change in vertical speed

Table 48. TCAS 7.0 Vertical Control

Data Type	Enum Value	Data Field Name	Description
int	0	VERT_CONT_NONE	None of the following vertical controls
int	1	CROSSING	Indicates that the ownship is crossing through the intruder's altitude.
int	2	REVERSE	Encounter in which it is necessary to reverse the original RA to avoid the threat
int	3	INCREASE	Increase in the strength of the initial RA to avoid a threat
int	4	MAINTAIN	RA requiring no change in vertical speed to avoid a threat

Table 49. TCAS 7.0 Up Advisory

Data Type	Enum Value	Data Field Name	Description
int	0	UP_ADV_NONE	No advisory
int	1	CLIMB	Climb @ 1500 to 2000 fpm
int	2	DONT_DESCEND	Do not descend
int	3	DONT_DESCEND500	Do not descend faster than 500 fpm
int	4	DONT_DESCEND1000	Do not descend faster than 1000 fpm
int	5	DONT_DESCEND2000	Do not descend faster than 2000 fpm

**Table 50. TCAS 7.0 Down Advisory**

Data Type	Enum Value	Data Field Name	Description
int	0	DOWN_ADV_NONE	No advisory
int	1	DESCEND	Descend @ 1500 to 2000 fpm
int	2	DONT_CLIMB	Do not climb
int	3	DONT_CLIMB500	Do not climb faster than 500 fpm
int	4	DONT_CLIMB1000	Do not climb faster than 1000 fpm
int	5	DONT_CLIMB2000	Do not climb faster than 2000 fpm

**Table 51. RA Output to Display and Aural Subsystem Every Cycle**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
float	rate	Feet per Minute (fpm)			Advisory rate to maintain	
int	combined_control				Enumerates the Combined Control Value	
int	vertical_control	UTC Decimal seconds			Enumerates the Vertical Control Value	
int	up_advisory	Decimal degrees	-90°	+90°	Enumerates the Up Advisory Value	
int	down_advisory	Decimal degrees	-180°	+180°	Enumerates the Down Advisory Value	
int	crossing				Enumerates the combined Control Value	1 = crossing 2 = not crossing

#### 4.3.18 TCAS 7.1 Message Structures

The FAA TCAS II v7.1 software application is integrated into the LVC system. The TCAS II 7.1 wrapper was developed by the NASA Langley SSI team while the original TCAS II 7.1 algorithm was provided by the FAA. The NASA Ames (ARC) IT&E team has modified the original wrapper to incorporate the different I/O infrastructure. The wrapper and the TCAS algorithm form the TCASProcessor application. The pertinent data structures and input dependencies are defined below. The TCASProcessor application generates traffic advisories (TA) and resolution advisories (RA) as part of the TCAS71 Message data structure. The intruderData array will vary in size due to the number of intruders that are picked up by the sensor range. The size of the TCAS71Message structure that is captured by the MsgSize in the MsgHeader will therefore dynamically change as intruder traffic is processed.

The MAXNUMBEROFTRAFFIC is set to a maximum value of 32.

**Table 52. TCAS 7.1 Message**

Data Type	Data Field Name	Units	Min	Max	Description	Notes
char	ownshipCallsign[ MAX_CALLSIGN_ LEN=8]				Ownship Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61. Maximum length = 8
Aural Alert	auralAlertLevel				Defines voice alert issued for displayed traffic alerts and RAs.	Enumerated in the AuralAlert table.
CombinedControl	combinedControl				Indicates the type of Resolution Advisory (RA)	Enumerated in the TCAS 7.1 Combined Control table.
int	escapeVSPDFPM	Feet per Minute (fpm)			Escape rate to maintain for safe separation	TCAS ZD Model
Down Advisory	downAdvisory				Defines value for downward sense alerts.	Enumerated in the TCAS 7.1 Down Advisory table.
Up Advisory	upAdvisory				Defines value for upward sense alerts.	Enumerated in the TCAS 7.1 Up Advisory table.
int	numberOfIntruders		0	32	Total number of intruders	MAXNUMBEROFTRAFFIC = 32
Tcas Intruder State	intruderData				Identification and Alert Level of intruder traffic	Defined by TCAS 7.1 Intruder State Message

**Table 53. TCAS 7.1 Intruder State**

Data Type	Data Field Name	Description	Notes
int	aircraftId	Computer identification of the intruder aircraft	
TcasAlertLevel	alertLevel	Defines the Alert Level of the intruder	Enumerated in the TCAS 7.1 Alert Level table

**Table 54. TCAS 7.1 Alert Level**

Data Type	Enum Value	Data Field Name	Description	Notes
int	0	NO_DATA	No alert data available	
int	1	NO_ALERT	No Alert status	
int	2	PROXIMITY_ALERT	Proximate Traffic - non-threat traffic within 6 nmi and $\pm 1200$ ft from the ownship.	Defined in TCAS 7.0 Threat table
int	3	TRAFFIC_ALERT_NO_ALT	Intruders that cause an TA to be issued (Mode A only)	
int	4	TRAFFIC_ALERT	Intruders that cause an TA to be issued (Mode A and C)	
int	5	RESOLUTION_ALERT	Intruders that cause an RA to be issued	

Table 55. TCAS 7.1 Aural Alert

Data Type	Value	Data Field Name	Description	Notes
int	0	AA_NONE	No Alerts	
int	1	AA_CLEAR	RA Removed	"Clear of Conflict"
int	2	MONITOR	Preventive RA	"Monitor Vertical Speed"
int	3	LEVEL_OFF	Reduce Climb or Descent RA, Weakening of RA	"Monitor Vertical Speed"
int	4	MAINTAIN_CROSSING	Altitude Crossing, Maintain Rate [of climb or descent] RA	"Maintain Vertical Speed, Crossing Maintain"
int	5	MAINTAIN	Maintain Rate [of climb or descent] RA	"Maintain Vertical Speed, Maintain"
int	6	INCREASE_DESCENT	Increase Descent RA	"Increase Descent, Increase Descent"
int	7	INCREASE_CLIMB	Increase Climb RA	"Increase Climb, Increase Climb"
int	8	DESCEND_NOW	RA Reversal to Descent RA	"Descend, Descend NOW; Descend, Descend NOW"
int	9	CLIMB_NOW	RA Reversal to Climb RA	"Climb, Climb NOW; Climb, Climb NOW"
int	10	DESCEND_CROSSING	Altitude Crossing Descend RA	"Descend, Crossing Descend; Crossing Descend"
int	11	CLIMB_CROSSING	Altitude Crossing Climb RA	"Climb, Crossing Climb; Crossing Climb"
int	12	DESCEND	Descend RA	"Descend, Descend"
int	13	CLIMB	Climb RA	"Climb, Climb"
int	14	NUM_AURAL_ALERTS	Number of Aural Alerts	

Table 56. TCAS 7.1 Combined Control

Data Type	Value	Data Field Name	Description
int	-999999	CC_NOT_SET	Combined Control not set
int	0	CC_NONE	No Advisory
int	1	CLEAR_OF_CONFLICT	Traffic conflict is resolved and RA is removed.
int	2	CLIMB_CORRECTIVE	RA requiring positive change in vertical speed
int	3	DESCEND_CORRECTIVE	RA requiring negative change in vertical speed
int	4	PREVENTIVE	RA requiring no change in vertical speed
int	PREVENTIVE	CC_MAX_ADVISORY	

**Table 57. TCAS 7.1 Down Advisory**

Data Type	Value	Data Field Name	Description	Notes
int	-999999	DOWN_ADVISORY_NOT_SET	Down Advisory not set	
int	0	DOWN_ADVISORY_NONE	No advisory	
int	1	DO_DESCEND	Descend @ specified rate	TCAS standard is 1500 to 2000 fpm
int	2	DONT_CLIMB	Do not climb faster than 250 fpm	
int	3	DONT_CLIMB500	Do not climb faster than 500 fpm	
int	4	DONT_CLIMB1000	Do not climb faster than 1000 fpm	
int	5	DONT_CLIMB2000	Do not climb faster than 2000 fpm	
int	= DONT_CLIMB2000	MAX_DOWN_ADVISORY	Do not climb faster than 2000 fpm	

**Table 58. TCAS 7.1 Up Advisory**

Data Type	Enum Value	Data Field Name	Description	Notes
int	-999999	UP_ADVISORY_NOT_SET	Up Advisory not set	
int	0	UP_ADVISORY_NONE	No advisory	
int	1	DO_CLIMB	Climb @ specified rate	TCAS standard is 1500 to 2000 fpm
int	2	DONT_DESCEND	Do not descend faster than 250 fpm	
int	3	DONT_DESCEND500	Do not descend faster than 500 fpm	
int	4	DONT_DESCEND1000	Do not descend faster than 1000 fpm	
int	5	DONT_DESCEND2000	Do not descend faster than 2000 fpm	
int	= DONT_DESCEND2000	MAX_UP_ADVISORY	Do not descend faster than 2000 fpm	

#### 4.3.19 Well Clear Recovery Message

The Well Clear Recovery Message is responsible for giving limited, suggestive guidance with guidance options in a particular direction, e.g. left turn, right turn, climb, and descend. Well clear recovery guidance has a mandatory requirement to inhibit altitude recovery guidance during a TCAS RA.

Well clear recovery guidance is engaged when the Detect and Avoid (DAA) guidance system (e.g. OmniBands) can no longer provide maneuvers guidance that remains well clear of an intruder, i.e. guidance is completely saturated with “non-green” guidance bands, or when a TCAS RA is generated against an intruder at the same time as a DAA Warning alert for a secondary intruder.

**Table 59. Well Clear Recovery Message**

Data Type	Data Field Name	Description	Notes
char	ownshipCallsign[ MAX_CALLSIGN_ LEN]	Ownship Call Sign	Call signs should conform to the FAA/ICAO requirements. Ex: SWA123, N123AB, SCAM61. Maximum length = 8
enum_Guidance Type_t	type	Defines the type of guidance (directional or limited suggestive)	Only the relevant field will be valid. Enumerated in the Guidance Type table.
enum_Directional Guidance_t	suggestion	Defines the recovery direction for either type, directional or limited suggestive.	Enumerated in the Directional Guidance table.
Range_t	range	Defines the recovery 'wedge'.	Defined if and only if the type is Limited Suggestive. If not, shown as null Enumerated in the Range table.

The enumerations and the Wedge data type are defined below.

**Table 60. Directional Guidance**

Data Type	Enum Value	Data Field Name	Description
int	0	TURN_LEFT	Generates a suggestion to turn left
int	1	TURN_RIGHT	Generates a suggestion to turn right
int	2	CLIMB	Generates a suggestion to climb
int	3	DESCEND	Generates a suggestion to descend
int	4	MAINTAIN	Generates a suggestion to remain at the present altitude and on present course.

**Table 61. Guidance Type**

Data Type	Enum Value	Data Field Name	Description
int	0	NONE	No guidance provided
int	1	DIRECTIONAL	Provides direction arrow indicators
int	2	LIMITED SUGGESTIVE	Provides the wedge indicating the recommend heading range for WCR.

**Table 62. Range Type**

Data Type	Enum Value	Data Field Name	Description
int	0	RELATIVE_HEADING	Heading WCR solution expressed in degrees relative to the ownship heading
int	1	ALTITUDE	Vertical WCR solution expressed in feet above MSL

*Note: Guidance represents recommended targets bounded by minimum and maximum bounds. The RangeType can be RELATIVE\_HEADING or ALTITUDE.*

**Table 63. Range**

Data Type	Data Field Name	Units	Description	Notes
enum_Range Type_t	target			Guidance target can be RELATIVE_HEADING or ALTITUDE
int	m_pad			
double	min	Altitude = Feet above MSL	Defines the minimum bounds of the WCR.	Relative Heading is determined in respect to ownship heading.
double	max	Relative Heading = Degrees	Defines the maximum bounds of the WCR.	

#### 4.4 Message Content Notes

##### 4.4.1 Heartbeat Message

The Gateway shall periodically send heartbeat message to the clients with enumeration defined below.

The LVC Gateway will send a periodic heartbeat message at a configurable time interval to every client for the sole purpose of detecting whether the client socket port has been closed. This infrastructure will detect a process that crashed and was running on the client connected to the Gateway. Upon detecting the closed socket, the Gateway will send MsgHeader message to every client unconditionally. The MsgHeader message requires no action, i.e. no response, by recipients. A message of type MsgHeader of size 12 bytes shall contain the value of 7030 in the MsgType field according to the definitions in Table 1. The MsgSize field (i.e. sizeof(MsgHeader)) shall be set to 12 bytes which essentially means there is no subsequent payload. Therefore, there is no need for the recipient to read the socket port of any further payload data.

Clients that receive the MsgHeader message shall be expected to consume this message nominally. Consequently, if the LVC Gateway does in fact detect a closed socket port, then it will forward a delete aircraft message, MsgDeleteAc, to all other active and valid subscribers. A MsgDeleteAc message shall be sent for each aircraft that was owned by the closed client.

##### 4.4.2 Primitive Data Type Definitions and Sizes in Bytes

The "C" structures displayed above are used on a Windows platform using x86 or x86-64 architecture. The byte order for Windows platforms is little endian (the least significant byte is stored first) and the sizes of the primitive data types are given below:

- long: 4 bytes
- unsigned long: 4 bytes
- int: 4 bytes
- unsigned int: 4 bytes
- short int: 2 bytes
- unsigned short int: 2 bytes
- char: 1 byte

- float: 4 bytes
- double: 8 bytes

#### 4.4.3 Byte Order and Need for Byte Swapping

All clients will publish messages in network byte order as computer networks transmit multi-byte numbers in this particular byte order. The most significant byte of a multi-byte number that is transmitted first over a network constitutes network byte order. Generally, different hosts (different CPUs) in the distributed environment can be little-endian or big-endian depending upon how bytes are ordered within a single word in the host memory. Therefore, when the little-endian host sends messages over the network it needs to convert (byte swap) them to network byte order before sending the messages out. Consequently, when the little-endian host receives a message over the network, it needs to convert the message back to host native byte representation, i.e. little-endian byte order.

## 5 Acronym List

ACAS-XU	Airborne Collision Avoidance System X Unmanned
ACFS	Advanced Concepts Flight Simulator
ACID	Aircraft Identification (call sign)
ADRS	Aeronautical Datalink and Radar Simulator
ADSB	Automatic Dependent Surveillance - Broadcast
AFRC	Armstrong Flight Research Center
API	Application Program Interface
ARC	Ames Research Center
ATC	Air Traffic Control
ATS	Air Traffic System
CA	Collision Avoidance
CAS	Calculated Airspeed
CATS	Change Action Tracking System
CDTI	Control Display Traffic Indicator
CI	Common Interface
CID	Computer Identification
CONOPS	Concept of Operations
CPA	Closest Point of Approach
CPDS	Conflict Prediction and Detection System
CSD	Cockpit Situation Display
CTAS	Center TRACON Automation System
DAA	Detect and Avoid
DPM	Deputy Project Manager
EPU	Estimated Position of Uncertainty
FAA	Federal Aviation Administration
FFSIM	Future Flight Simulator
FIS-B	Flight Information Services-Broadcast
FL	Flight Level
FLAPS	Flexible Acquisition Processing System
FPM	Feet per minute
FPS	Feet per second
FRD	Fix, Radial, Distance
FS	Flight State
FT	Flight Test
GCS	Ground Control Station
HFOM	Horizontal Figure of Merit
HFOMR	Horizontal Figure of Merit - Rate (Velocity)
HHMMSS	Hours, Minutes, Seconds
HIL	Horizontal Integrity Limit
HITL	Human in the Loop
HLA	High Level Architecture
HVE	Horizontal Velocity Error
ICAO	International Civil Aviation Organization
ICD	Interface Control Document

IHITL	Integrated Human in the Loop
IP	Internet Protocol
IT&E	Integrated Testing and Evaluation
LaRC	Langley Research Center
LMA	Link Management Assembly
LNAV	Lateral Navigation
LoWC	Loss of Well Clear
LVC	Live Virtual Constructive
MACS	Multi Aircraft Control Simulation
MOPS	Minimum Operational Standards
MPI	Multipurpose Protocol Interface
MSL	Mean Sea Level
NAS	National Airspace System
NASA	National Aeronautics and Space Administration
NM	Nautical Miles
PROX	Proximal, Proximity
PT	Part Task
RA	Resolution Advisory
RFDP	Remote Flight Data Processor
RGCS	Research Ground Control Station
RUMS	Remote Users Monitor Service
RWC	Remain Well Clear
SAA	Sense and Avoid
SaaProc	Sense And Avoidance Processor
SAIC	Science Applications International Corporation
SRD	Systems Requirement Document
SS	Self-Separation
STM	Surveillance and Tracking Module (ACAS-Xu)
SWRD	Software Requirement Document
TA	Traffic Advisory
TAS	True Airspeed
TCAS	Traffic Collision Avoidance System
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TISB	Traffic Information Service – Broadcast
TRM	Threat Resolution Module (ACAS-Xu)
UAS	Unmanned Aircraft System
UASRP	Unmanned Aircraft Systems Research Platform
UAT	Universal Access Transceiver
UDP	User Datagram Protocol
UTC	Coordinated Universal Time
VAST-	Virtual Airspace Simulation Technology
HLA	High Level Architecture
VFR	Visual Flight Rules
VHF	Very High Frequency
VNAV	Vertical Navigation
VPN	Virtual Private Network

VSCS	Vigilant Spirit Control Station
Vspd	Vertical Speed
WCR	Well Clear Recovery

## 6 Appendix A. Detect and Avoid Symbology

Symbol	Name	<u>Pilot Action</u>	Aural Alert Verbiage
	TCASRA (Cooperative Only)	<ul style="list-style-type: none"> <li>• <b>Immediate action required</b></li> <li>• No researcher coordination required</li> </ul>	"Climb/Descend"
	DAA Warning Alert	<ul style="list-style-type: none"> <li>• <b>Immediate action required</b></li> <li>• No researcher coordination required</li> </ul>	"Traffic, Maneuver Now"
	Corrective DAA Alert	<ul style="list-style-type: none"> <li>• On current course, <b>corrective action required</b></li> <li>• Coordinate maneuver with researcher prior to maneuvering</li> </ul>	"Traffic, Avoid"
	Preventive DAA Alert	<ul style="list-style-type: none"> <li>• <b>No action required</b></li> <li>• No coordination required</li> </ul>	"Traffic, Monitor"
	None (Target)	<ul style="list-style-type: none"> <li>• <b>No action required</b></li> <li>• No coordination required</li> </ul>	N/A

## 7 Appendix B. ADRS Proc and Category Values

Table B1. ADRS\_proc Values

<u>Data Origin</u>	<u>Value</u>	<u>Sources</u>
ADRS_PROC_UNKNOWN	0	Unknown source
ADRS_PROC_PAS	1	Psuedo Aircraft Simulator
ADRS_PROC_PAS_ADSB	2	Psuedo Aircraft Simulator, ADSB
ADRS_PROC_PAS_FFSIM	3	Psuedo Aircraft Simulator, Future Flight Simulation
ADRS_PROC_PAS_CI	4	Psuedo Aircraft Simulator, Common Interface
ADRS_PROC_FSR_GENERIC	10	Generic Flight Simulator Report
ADRS_PROC_FSR_ACFS	11	Advanced Concept Flight Simulator Flight Simulator Report
ADRS_PROC_FSR_B747	12	B747 Flight Simulator Report
ADRS_PROC_FSR_CDTI	13	Control Display Traffic Indicator Flight Simulator Report
ADRS_PROC_FSR_ASTOR	14	Astor Flight Simulator Report
ADRS_PROC_FSR_PC_PLANE	15	Microsoft Flight Simulator Flight Simulator Report
ADRS_PROC_FSR_B757	16	B757 Flight Simulator Report (corresponds to old PROC_PACS)
ADRS_PROC_FSR_MINIACFS	17	Custom ACFS Flight Simulator Report
ADRS_PROC_ISM_C	20	
ADRS_PROC_ISM_T	25	
ADRS_PROC_MPI_CLIENT	30	LVC Gateway
ADRS_PROC_MPI_CLIENT_ADRS	31	LVC Gateway, ADRS
ADRS_PROC_MPI_CLIENT_PILOT	32	LVC Gateway, MACS Pilot Position
ADRS_PROC_MPI_CLIENT_CATS	33	LVC Gateway, Change Action Tracking System
ADRS_PROC_MPI_CLIENT_FREQ_CENTER	34	LVC Gateway, Frequency Center
ADRS_PROC_MPI_CLIENT_MACS	35	LVC Gateway, Multi Aircraft Control System
ADRS_PROC_MPI_CLIENT_ATS_GATEWAY	36	LVC Gateway, Application Tracking System Gateway
ADRS_PROC_MPI_CLIENT_CTAS	37	LVC Gateway, Center TRACON Automation System
ADRS_PROC_MPI_CLIENT_VAST	38	LVC Gateway, Virtual Airspace Simulation Technologies
ADRS_PROC_MPI_SERVER	40	LVC Gateway as server
ADRS_PROC_MPI_SERVER_ADRS	41	LVC Gateway as server with ADRS
ADRS_PROC_CI_CLIENT	50	Common Interface
ADRS_PROC_CI_CLIENT_ADRS	51	Common Interface, ADRS
ADRS_PROC_CI_CLIENT_CDTI	52	Common Interface, Control Display Traffic Indicator

**Table B2. Category Values**

<u>Flight Status</u>	<u>Value</u>
ADRS_MPI_DEPARTURE	1
ADRS_MPI_ARRIVAL	2
ADRS_MPI_OVERFLIGHT	3
ADRS_MPI_SATELLITE_ARRIVAL	4
ADRS_MPI_SATELLITE	5
ADRS_MPI_DEPARTURE_ARRIVAL	6
ADRS_MPI_SATELLITE_DEPARTURE	7
ADRS_MPI_SATELLITE_TO_EXTERNAL	8
ADRS_MPI_EXTERNAL_TO_SATELLITE	9
ADRS_MPI_AMBIGUOUS_CATEGORY	4096

THIS PAGE INTENTIONALLY LEFT BLANK