Antenna Interface Standards Group

Subunit Type Standard AISG-ST-ALS

vALS3.0.2.2

27th June 2024

Antenna location and orientation sensor
AISG-ST-ALS
vALS3.0.2.2

Revision History

DATE	ISSUE	NOTES
June 27th, 2024	vALS3.0.2.2	Third public release
June 29th, 2023	vALS3.0.1.3	Second public release
January 31st, 2022	vALS3.0.0.4	First public release

1. FOREWORD (Informative)

This standard has been produced by the Antenna Interface Standards Group (AISG) to introduce and define new features and enhancement of the management system for antenna line devices (ALDs) with remote control and monitoring facilities. AISG v3.0 base standard describes the common behaviour of antenna line devices with AISG interfaces and type-specific functionality is defined in subunit type standards.

This subunit type standard defines the antenna location and orientation sensor (ALS) subunit type. This subunit is able to provide the following information from sensor or previously stored data.

- Geographical location of an antenna (called *coordinate sensor* capability).
- Altitude of an antenna (called *altitude sensor* capability).
- Azimuth, tilt and roll of an antenna (called *orientation sensor* capability).

ALS subunit also has the following additional features:

- If there is no corresponding sensor (that is, a capability is not supported), store and return the corresponding data. An ALS shall support at least one capability.
- Ability to generate alarms if the orientation of the antenna monitored by ALS changes outside of the set orientation alarm limits.
- Ability to record the minimum and maximum antenna orientation values since the last time orientation monitor values were cleared.

This standard is independent of previous 3GPP specifications.

2. SCOPE (Informative)

AISG v3.0 specifies the standard data interface between a primary, typically a base station, and antenna line devices (ALDs) which are manageable units, usually associated with base station antennas.

The standard is divided into the base standard and several subunit type standards. This subunit type standard document describes the specific behaviour of the Antenna Location and orientation Sensor (ALS) subunit type.

This standard defines the functional behaviour of ALS subunits.

2.1. Interpretation (Normative)

The text of the standard defines explicitly what is required or permitted. Anything that is not explicitly allowed is not permitted.

All statements in this document are normative, unless indicated as informative, notes or as an example.

For purposes of compliance and AISG interoperability, users should note that the implementation of this subunit type standard is optional. However, once it is selected for inclusion in a product, the entire standard becomes mandatory.

3. REFERENCES

This AISG Standard incorporates provisions from other publications. These are cited in the text and the referenced publications are listed below. Where references are listed with a specific version or release, subsequent amendments or revisions of these publications apply only when specifically incorporated by amendment or revision of this AISG standard. For references listed without a version or release, the latest edition of the publication referred to applies.

1 AISG v3.0: “Base Standard AISG v3.0”

2 AISG v3.0 STCM: “Subunit Type Compliance Matrix”

3 NIMA TR8350.2 U.S. Department of Defence World Geodetic System 1984, Third Edition – Amendment 1

4. VERSION COMPLIANCE (Informative)

The compliance of this standard with different version of AISG v3 baseline standard is defined in [2].

5. ABBREVIATIONS (Informative)

Where abbreviations or acronyms are used in this document they have the following meanings:

ALD	Antenna Line Device
ALS	Antenna Location and orientation Sensor
MALD	Multi-primary ALD
SALD	Single-primary ALD
TCC	Time-Consuming Command
3GPP	3rd Generation Partnership Project
WGS	World Geodetic System

6. TERMINOLOGY

Where the following terms are used in this document, they have the following meanings:

Altitude	The elevation of a point or object from the surface of the WGS84 [3] geoid. Altitude is reported in meters, to one decimal place in accuracy.
Altitude precision	A value that represents the standard deviation of values from its arithmetic mean measured by the ALS. This value may be used to judge the credibility of the values from the relevant sensor. Precision is reported in meters, to one decimal place of accuracy.
Measured altitude	A value representing the “best estimate” by the ALS of the altitude. This value may be the result of averaging historical data, or any other means of processing deemed appropriate by the ALS vendor for producing a consistently accurate result.
Measured location	Coordinates representing the “best estimate” by the ALS (of the latitude and the longitude). This value may be the result of averaging historical data, or any other means of processing deemed appropriate by the ALS vendor for producing a consistently accurate result.
Measured mechanical azimuth / tilt / roll	A value representing the “best estimate” of an antenna in one of three axes measured by the ALS (azimuth, tilt, roll). This value is the result of averaging historical data, or any other means of processing deemed appropriate by the ALS vendor for producing a consistently accurate result.
Electrical tilt	The electrical tilt angle is, in the vertical cut, the angle between the antenna mechanical boresight and the half-power beam axis. Positive tilt values represent main beam position below boresight. Negative tilt values represent main beam position above boresight.
Half-power beam axis	The half-power beamwidth is, in a radiation pattern cut containing the beam peak axis, the angle between the two closest directions in which the radiation intensity is one-half the maximum value; its bisect is called half-power beam axis.
Highest monitored mechanical azimuth / tilt / roll	A value that represents the highest measured result in one of the three axes measured by the ALS (azimuth, tilt, roll) since orientation monitoring values were last cleared. It may be used to analyse the effect of sway or other mechanical disturbances.
Latitude	An angular measurement in degrees ranging from 0 degrees at the equator, to +90 degrees at the North pole, and −90 degrees at the South pole, as specified by WGS84 [3].

	Latitude identifies a position on earth north or south of the equator.
Location precision	A value that represents the standard deviation of values from their arithmetic mean in one of two dimensions measured by the ALS (latitude and longitude). This value may be used to judge the credibility of the values from the relevant sensor. Precision is reported in meters, to one decimal place of accuracy.
Longitude	An angular measurement ranging from 0 degrees at the prime meridian (Greenwich meridian) to +180 degrees eastward and −180 degrees westward, as specified by the WGS84 [3] ellipsoid. Longitude identifies a position on earth east or west of the prime meridian.
Lowest monitored mechanical azimuth / tilt / roll	
	A value that represents the lowest measured result in one of the three axes measured by the ALS (azimuth, tilt, roll) since the orientation monitoring values were last cleared. It may be used to analyse the effect of sway or other mechanical disturbances.
Mechanical azimuth	Heading angle of the ALS in the horizontal plane. In the case of directional antenna, the ALS is referenced to the boresight of the physical antenna assembly. The mechanical azimuth is equal to the direction of RF propagation when zero electrical tilt, azimuth steering, or phase shifting has been applied.
Mechanical boresight	The axis perpendicular with the antenna aperture.
Mechanical roll	Roll angle of the ALS in the vertical plane between the vertical and physical axes of the antenna. In the case of a directional antenna, while facing in the direction of mechanical azimuth, roll at an angle clockwise from the vertical shall be represented by a positive number, while roll in the counter-clockwise direction shall be represented by a negative number.
Mechanical tilt	Tilt angle of the ALS in vertical plane. Tilt at an angle below straight and level shall be represented by a positive number (down-tilt), while tilt at an angle above straight and level shall be represented by negative number.
Orientation	Orientation comprises tilt, roll and azimuth.
Target mechanical azimuth / tilt / roll	
	A value representing the target alignment of an antenna in one of three axes measured by the ALS (azimuth, tilt, roll).

7. DEFINITIONS

```
Enumeration ALSAlarmCode_t : uint16_t {
     ALSAlarmMechanicalTilt    ← 0x0400
     ALSAlarmMechanicalRoll    ← 0x0401
     ALSAlarmMechanicalAzimuth ← 0x0402
}

CONSTANT uint16_t NrOfSubunitAlarms ← 3 // Number of subunit alarm types for this
subunit type

// Global variables for orientation monitoring // change to degree type from base
degree_t LowestMonitoredMechanicalTilt         ← +3276.7  // Indicates a NULL value
degree_t HighestMonitoredMechanicalTilt        ← +3276.7  // That is, orientation
degree_t LowestMonitoredMechanicalRoll         ← +3276.7  // monitoring has not yet
degree_t HighestMonitoredMechanicalRoll        ← +3276.7  // recorded any value for
degree_t LowestMonitoredMechanicalAzimuth      ← +3276.7  // that monitored
degree_t HighestMonitoredMechanicalAzimuth     ← -3276.7  // parameter
```

8. GENERAL ASPECTS

8.1. ALS general description

An implementation of an ALS subunit may contain one or more of the following capabilities:

- Geographical location of an antenna (called *coordinate sensor* in this standard)
- Altitude of an antenna (called *altitude sensor* in this standard)
- Azimuth, tilt and roll of an antenna (called *orientation sensor* in this standard)

If the ALS subunit supports a capability, it contains an on-board sensor that is able to measure information related to that capability, and is able to report this information to the primary.

If a capability is not supported, the ALS subunit is able to report related information previously stored to the ALS subunit to the primary.

This standard defines a command for the primary to query which of these capabilities are supported by a particular implementation of the ALS subunit.

8.1.1. Orientation monitoring

The orientation function capability includes orientation monitoring that monitors and records the lowest and highest orientation values (tilt, roll and azimuth) measured since these values were last cleared. These values are cleared during an ALD reset. The command ALSGetMonitoredOrientationInformation can also be used to clear these values.

It is recommended that the ALS vendor uses countermeasures to stop noise and short disturbances in the measured orientation values impacting the orientation monitoring. For example by employing methods such as filtering or averaging the measured values.

8.1.2 Orientation alarm

The orientation sensor capability provides orientation alarms.

The orientation alarms are raised separately for tilt, roll and azimuth when the measured orientation values are outside the corresponding orientation alarm window specified by the orientation alarm limits.

The alarm window for mechanical tilt alarm is specified as follows: The range of lower and upper alarm limits is from −90.0° to -90.0°. The lower limit shall be lower than the upper limit.

The mechanical tilt alarm shall not be raised if the provenance of one or both the upper and lower limits are NotSet.

The alarm window for mechanical roll alarm is specified as follows: The range of lower and upper alarm limit is from −180.0° to -180.0°. The lower limit has to be lower than the upper limit.

The mechanical roll alarm shall not be raised if the provenance of one or both the upper and lower limits are NotSet.

The alarm window for the mechanical azimuth is specified as follows: The range of the limit1 and limit2 is from 0° to 359.9°. Limit1 and limit2 cannot be the same.

The mechanical azimuth alarm shall not be raised if the provenance of one or both limit1 and limit2 are NotSet.

See Annex B for use case examples of orientation alarm limits.

8.2. Subunit relationship

An ALS subunit has a logical relationship with array element(s) of an antenna.

8.3. State models

This subunit type standard does not have state models.

8.4. ALS capabilities

An ALS may- have sensors for geographical location, altitude and orientation. If present, each of these supports a capability.

The ALSCapabilities are represented by the following bitfield and are set by design.

```
Bitfield ALSCapabilities_t : uint8_t {
     CoordinateSensor               : Bit 0
     AltitudeSensor                 : Bit 1
     OrientationSensor              : Bit 2
}

ALSCapabilities_t ALSCapabilities
```

At least one capability must be supported.

Bit value 0: Capability not supported.

Bit value 1: Capability supported.

If the capability is supported, the ALS provides data read from the related sensor. If the capability is not supported, the ALS provides storage for the related data. Setting the related data is only allowed if the capability is not supported. If the capability is supported, the provenance shall be Automatic, otherwise it shall be Manual.

8.5. Return codes

This subunit type standard extends the following subunit type-specific return codes.

```
Enumeration ReturnCode_t : uint16_t {
     ALSSensorDataNotAvailable ← 0x0400
}
```

8.6. Resumption of operation

The following data shall be retained after reset:

- Location information (if applicable)
- Altitude information (if applicable)
- Orientation information (if applicable)

8.7. Default values when shipped from the factory

When shipped from the factory, the ALS subunit shall contain the data presented in sections 8.7.1, 8.7.2, and 8.7.3.

8.7.1. Defaults for ALSSetCoordinates, ALSSetAltitude and ALSSetOrientation

The ALSSetCoordinates, ALSSetAltitude and ALSSetOrientation commands can only be executed if the corresponding capability is not supported. Therefore the default values for these commands can only be set when the corresponding capability **is not** supported.

ALSSetCoordinates default values			**ALSSetAltitude default values**			**ALSSetOrientation default values**	
Parameter	**Default Data**		**Parameter**	**Default data**		**Parameter**	**Default data**
Latitude	0		Altitude	0		MechanicalTilt	0
Latitude Provenance	NotSet		Altitude Provenance	NotSet		MechanicalTilt Provenance	NotSet
Longitude	0		AltitudePrecision	0		MechanicalRoll	0
Longitude Provenance	NotSet		AltitudePrecision Provenance	NotSet		MechanicalRoll Provenance	NotSet
LocationPrecision	0					MechanicalAzimuth	0
LocationPrecision Provenance	NotSet					MechanicalAzimuth Provenance	NotSet

Table 8.7.1-1: Defaults for ALSSetCoordinates, ALSSetAltitude and ALSSetOrientation

8.7.2. Defaults for ALSSetOrientationAlarmLimits

The orientation alarm feature requires that the ALS subunit supports the orientation sensor capability. Orientation alarm limits are used to define the alarm limits of the orientation alarms. Therefore, default values for orientation alarm limits can only be set when the orientation sensor capability **is** supported.

ALSSetOrientation AlarmLimits Default alarm limit values for tilt			**ALSSetOrientation AlarmLimits Default alarm limit values for roll**			**ALSSetOrientation AlarmLimits Default alarm limit values for azimuth**	
Parameter	**Default Data**		**Parameter**	**Default data**		**Parameter**	**Default data**

LowerMechanical TiltAlarmLimit	0		LowerMechanical RollAlarmLimit	0		Mechanical AzimuthAlarmLimit1	0
LowerMechanical TiltAlarmLimit Provenance	NotSet		LowerMechanical RollAlarmLimit Provenance	NotSet		Mechanical AzimuthAlarmLimit1 Provenance	NotSet
UpperMechanical TiltAlarmLimit	0		UpperMechanical RollAlarmLimit	0		Mechanical AzimuthAlarmLimit2	0
UpperMechanical TiltAlarmLimit Provenance	NotSet		UpperMechanical RollAlarmLimit Provenance	NotSet		Mechanical AzimuthAlarmLimit2 Provenance	NotSet

Table 8.7.2-1: Defaults for ALSSetOrientationAlarmLimits

8.7.3. Defaults for ALSSetArrayElementNumbersToSubunit

The default association between array element numbers and subunits is vendor specific. If the ALS is shipped from the factory with no array elements associated to the ALS subunit, the ALS subunit shall contain the data presented in the table 8.7.3-1 below.

Parameter	Default value
NrOfArrayElements	0
ArrayElementNumbers[]	Empty list
ArrayElementNumbers Provenance	NotSet

Table 8.7.3-1: Defaults for ALSSetArrayElementNumbersToSubunit

9. LAYER 1

All definitions and specifications for ALDs in [1] regarding layer 1 shall be valid for ALDs which contain ALS subunits.

9.1. DC power consumption

This subunit type standard does not define the power consumption of a ALS. See [1] for details on how power consumption is reported to the primary.

This subunit type standard does not contain any commands that switch the ALD from SteadyStatePowerMode to HighPowerMode or to SleepMode.

10. LAYER 2

All definitions and specifications for ALDs in [1] regarding layer 2 shall be valid for ALDs which contain ALS subunits.

11. LAYER 7

An ALD which contains ALS subunit(s) shall support the command set which is defined in [1] in addition to those commands specified in this standard.

11.1. Subunit type

Subunit type	1-octet unsigned integer code
ALS	0x04

Table 11.1-1: Subunit type code

11.2. Overview of commands for ALS subunits

The table below shows an overview of all commands used in this ALS subunit type standard.

The following abbreviations are used in the Table 11.2-1: “Commands for ALS subunits”

M Mandatory

O Optional

- Not applicable

	Code	Initiator	Subunit	Timeout	TCC	Mandatory for:			Changes the ConnectionState	Changes to HighPowerMode	Minimum require authority
						Primary	SALD	MALD			
ALS commands											
ALS Get Capabilities	0x0400	Primary	>0	1 s	no	M	M	M	no	no	RO
ALS Set Coordinate	0x0401	Primary	>0	1 s	no	M	M	M	no	no	RW
ALS Get Coordinate	0x0402	Primary	>0	1 s	no	M	M	M	no	no	RO
ALS Set Altitude	0x0403	Primary	>0	1 s	no	M	M	M	no	no	RW
ALS Get Altitude	0x0404	Primary	>0	1 s	no	M	M	M	no	no	RO
ALS Set Orientation Alarm Limits	0x0405	Primary	>0	1 s	no	M	M	M	no	no	RW
ALS Get Orientation Alarm Limits	0x0406	Primary	>0	1 s	no	M	M	M	no	no	RO
Reserved 1	0x0407	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
Reserved 2	0x0408	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
ALS Get Orientation Information	0x0409	Primary	>0	1 s	no	M	M	M	no	no	RO
ALS Get Monitored Orientation Information	0x0410	Primary	>0	1 s	no	M	M	M	no	no	RO
Site mapping command for ALS											

	Code	Initiator	Subunit	Timeout	TCC	Mandatory for:			Changes the ConnectionState	Changes to HighPowerMode	Minimum require authority
						Primary	SALD	MALD			
ALS Set Array Element Numbers To Subunit	0x0411	Primary	>0	1 s	no	O	M	M	no	no	RW
ALS Get Array Element Numbers Of Subunit	0x0412	Primary	>0	1 s	no	O	M	M	no	no	RO

Table 11.2-1: Commands for ALS subunits

11.3. Altitude representation

Altitude is reported in meters, to one decimal place in accuracy, and then multiplied by 10 so that it may represented by an integer in 0.1 m units with a range of -10000 to 100000.

Altitude precision is reported in meters, to one decimal place in accuracy, and then multiplied by 10 so that it may represented by an integer in 0.1 m units with a range of 0 to 1000.

11.4. Mechanical azimuth representation

The mechanical azimuth range supported is from 0.0° to +359.9° east of geographical true North. For example azimuth 0.0° represents North and 90.0° represents East (see Annex A). The azimuth value is expressed in 0.1° units with a range of 0 to +3599.

11.5. Mechanical tilt representation

The mechanical tilt range supported is from −90.0° to +90.0°. Positive tilt values represent downtilt and negative tilt values represent uptilt (see Annex A). The tilt value is expressed in 0.1° units with a range of −900 to +900.

11.6. Mechanical roll representation

The mechanical roll range supported is from −180.0° to +180.0°. Positive values represent clockwise rotation of the antenna seen from behind. A roll angle of 0.0° represents an antenna oriented vertically (see Annex A). The roll value is expressed in 0.1° units with a range of −1800 to +1800.

11.7. Latitude representation

The latitude is reported in decimal degrees, to six decimal places of accuracy, and then multiplied by 1×10^6 so that it may represented by an integer, with a range of -90000000 to +90000000.

11.8. Longitude representation

The longitude is reported in decimal degrees, to six decimal places of accuracy, and then multiplied by 1×10^6 so that it may represented by an integer, with a range of -180000000 to +180000000.

11.9. Location precision representation

The location precision is reported in meters, to one decimal places of accuracy, and then multiplied by 10 so that it may represented by an integer, with a range of 0 to +10000.

11.10. Alarm handling

After a reset, all alarm states shall be cleared.

After ClearActiveAlarms command is received from a primary, all alarm states shall be cleared. An ALS subunit can also clear its alarms autonomously.

The alarm logic, including the time over which the parameter has to exceed its threshold value before an alarm is raised, is vendor specific. The same is true for the clearing of the alarm.

See Section 11.11 “Monitoring events” for pseudocode handling alarm raising.

11.11. Monitoring events

```
// Handling alarm events

IF ALSCapabilities.OrientationSensor THEN
    IF (LowerMechanicalTiltAlarmLimit < UpperMechanicalTiltAlarmLimit) AND
    LowerMechanicalTiltAlarmLimitProvenance NOT NotSet AND
    UpperMechanicalTiltAlarmLimitProvenance NOT NotSet THEN

        ON «the ALS detects exceeding of mechanical tilt alarm limits»
            RAISE ALSAlarmMechanicalTilt SEVERITY Major ON Cmd.Subunit
        DONE
    ENDIF

    IF (LowerMechanicalRollAlarmLimit < UpperMechanicalRollAlarmLimit) AND
    LowerMechanicalRollAlarmLimitProvenance NOT NotSet AND
    UpperMechanicalRollAlarmLimitProvenance NOT NotSet THEN
        ON «the ALS detects exceeding of mechanical roll alarm limits»
            RAISE ALSAlarmMechanicalRoll SEVERITY Major ON Cmd.Subunit
        DONE
    ENDIF

    IF (MechanicalAzimuthAlarmLimit1 ≠ MechanicalAzimuthAlarmLimit2) AND
    MechanicalAzimuthAlarmLimit1Provenance NOT NotSet AND
    MechanicalAzimuthAlarmLimit2Provenance NOT NotSet THEN
        IF MechanicalAzimuthAlarmLimit1 > MechanicalAzimuthAlarmLimit2 THEN
            ON «Measured mechanical azimuth» < MechanicalAzimuthAlarmLimit1
            AND «Measured mechanical azimuth» > MechanicalAzimuthAlarmLimit2
                RAISE ALSAlarmMechanicalAzimuth SEVERITY Major
                ON Cmd.Subunit
            DONE
        ELSE
            ON «Measured mechanical azimuth» < MechanicalAzimuthAlarmLimit1
            OR «Measured mechanical azimuth» > MechanicalAzimuthAlarmLimit2
                RAISE ALSAlarmMechanicalAzimuth SEVERITY Major
                ON Cmd.Subunit
            DONE
        ENDIF
    ENDIF
ENDIF

// Handling orientation monitoring events

IF ALSCapabilities.OrientationSensor THEN
    ON «ALS measured tilt value» < LowestMonitoredMechanicalTilt
        LowestMonitoredMechanicalTilt ← «ALS measured tilt value»
    DONE
    ON «ALS measured tilt value» > HighestMonitoredMechanicalTilt
        HighestMonitoredMechanicalTilt ← «ALS measured tilt value»
    DONE
    ON «ALS measured roll value» < LowestMonitoredMechanicalRoll
        LowestMonitoredMechanicalRoll ← «ALS measured roll value»
```

DONE
ON «ALS measured roll value» > HighestMonitoredMechanicalRoll
HighestMonitoredMechanicalRoll ← «ALS measured roll value»
DONE
ON «ALS measured azimuth value» < LowestMonitoredMechanicalAzimuth
LowestMonitoredMechanicalAzimuth ← «ALS measured azimuth value»
DONE
ON «ALS measured azimuth value» > HighestMonitoredMechanicalAzimuth
HighestMonitoredMechanicalAzimuth ← «ALS measured azimuth value»
DONE
ENDIF

// Handling orientation monitoring value clearing by Reset ALD

ON «ALS received Reset ALD command»
LowestMonitoredMechanicalTilt ← +3276.7
HighestMonitoredMechanicalTilt ← +3276.7
LowestMonitoredMechanicalRoll ← +3276.7
HighestMonitoredMechanicalRoll ← +3276.7
LowestMonitoredMechanicalAzimuth ← +3276.7
HighestMonitoredMechanicalAzimuth ← +3276.7
DONE

NOTE: The value +3276.7 indicates that no value has been set to that parameter.

11.12. ALS commands

11.12.1. ALS Get Capabilities

Description (Informative):

On receipt of this command the ALS subunit shall return the ALS capabilities:

- ALS subunit supports coordinate sensor capability (TRUE/FALSE)
- ALS subunit supports altitude sensor capability (TRUE/FALSE)
- ALS subunit supports orientation sensor capability (TRUE/FALSE)

Primaries should use this command whenever the ALD has performed a reset.

Message format:

```
PrimaryCommand ALSGetCapabilitiesCommand {
      CommandCode_t         Command ← 0x0400
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}
```

```
ALDResponse ALSGetCapabilitiesResponse {
      CommandCode_t         Command ← 0x0400
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            ALSCapabilities_t ALSCapabilities
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t {
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      OK
}
```

Primary pseudocode:

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD
      AND ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
      RETURN NotAuthorised
      EXIT
ENDIF

result ← IsCommandAllowed( LIST{      OperatingConnectionState
                                      RestrictedConnectionState},
                                      Cmd.Command, CurrentPort)

UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF

RETURN OK, ALSCapabilities
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.12.2. ALS Set Coordinates

Description (Informative):

On receipt of this command the ALS subunit shall store the external geographic location information and their precision received from primary to non-volatile memory. If CoordinateSensor capability is supported, the command is rejected, and the ALS subunit returns DataReadOnly return code.

Message format:

```
PrimaryCommand ALSSetCoordinatesCommand {
      CommandCode_t         Command ← 0x0401
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      degree32_t            Latitude
      Provenance_t          LatitudeProvenance
      degree32_t            Longitude
      Provenance_t          LongitudeProvenance
      meter_t               LocationPrecision
      Provenance_t          LocationPrecisionProvenance
}

ALDResponse ALSSetCoordinatesResponse {
      CommandCode_t         Command ← 0x0401
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t {
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      DataReadOnly
      OutOfRange
      InvalidProvenance
      GeneralError
      OK
}
```

Primary pseudocode:

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
    UNLESS ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = ReadWrite THEN
        RETURN NotAuthorised
        EXIT
    ENDIF
ENDIF

result ← IsCommandAllowed( LIST{    OperatingConnectionState,
                                    RestrictedConnectionState},
                                    Cmd.Command, CurrentPort)

UNLESS result.allowed THEN
    RETURN result.code
    EXIT
ENDIF

IF ALSCapabilities.CoordinateSensor THEN
    RETURN DataReadOnly
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.Latitude < -90 degree32_t OR Cmd.Latitude > 90 degree32_t THEN
    RETURN OutOfRange
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.Longitude < -180 degree32_t OR Cmd.Longitude > 180 degree32_t THEN
    RETURN OutOfRange
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.LocationPrecision > 10 meter_t THEN
    RETURN OutOfRange
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.LatitudeProvenance NOT Manual THEN
    RETURN InvalidProvenance
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.LongitudeProvenance NOT Manual THEN
    RETURN InvalidProvenance
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
```

ENDIF

IF Cmd.LocationPrecisionProvenance NOT Manual THEN
    RETURN InvalidProvenance
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

«store the longitude and latitude information, their provenances and precisions to non-volatile memory»
IF «the ALD detects a hardware error» THEN
    // Replace “Hardware error” with descriptive text to be read using
    // GetDiagnosticInformation
    RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
    RETURN “GeneralError”
ELSE
    RETURN OK
ENDIF

CommandExit(Cmd.Command, CurrentPort)
EXIT

11.12.3. ALS Get Coordinates

Description (Informative):

On receipt of this command the ALS subunit shall return the measured location information from the on-board sensor, its provenances and precision. If CoordinateSensor capability is not supported, the information set by ALS Set Coordinates command is returned.

Message format:

```
PrimaryCommand ALSGetCoordinatesCommand {
     CommandCode_t         Command ← 0x0402
     CommandSequence_t     PrimaryCommandSequence
     Subunit_t             Subunit
     DataLength_t          DataLength ← 0
}
```

```
ALDResponse ALSGetCoordinatesResponse {
      CommandCode_t         Command ← 0x0402
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            BOOLEAN         LocationProvidedBySensor
            degree32_t      Latitude
            Provenance_t    LatitudeProvenance
            degree32_t      Longitude
            Provenance_t    LongitudeProvenance
            meter_t         LocationPrecision
            Provenance_t    LocationPrecisionProvenance
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t{
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      OK
      ALSSensorDataNotAvailable
}
```

Primary pseudocode):

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
      IF ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
            RETURN NotAuthorised
            EXIT
      ENDIF
ENDIF
result ← IsCommandAllowed( LIST{    OperatingConnectionState,
                                    RestrictedConnectionState},
                                    Cmd.Command, CurrentPort)
```

```
UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF

// Coordinate sensor capability supported, data coming from the on-board sensor
IF ALSCapabilities.CoordinateSensor THEN
      LocationProvidedBySensor ← TRUE
      LatitudeProvenance ← Automatic
      LongitudeProvenance ← Automatic
      LocationPrecisionProvenance ← Automatic
ELSE
      LocationProvidedBySensor ← FALSE
ENDIF

IF ALSCapabilities.CoordinateSensor THEN          // Data from sensor
      IF «Location sensor data available» THEN
            RETURN OK, LocationProvidedBySensor, «Measured latitude, longitude,
            location precision and corresponding provenances from the sensor»
      ELSE
            RETURN ALSSensorDataNotAvailable
            CommandExit(Cmd.Command, CurrentPort)
            EXIT
      ENDIF
ELSE                                              // Data from memory
      RETURN OK, LocationProvidedBySensor, «Latitude, longitude, location precision
      and corresponding provenances previously stored by ALSSetCoordinates»
ENDIF     // If the ALS was shipped from the factory with no data in these fields, the
            // corresponding provenances are NotSet

CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.12.4. ALS Set Altitude

Description (Informative):

On receipt of this command the ALS subunit shall store the altitude information and its precision received from primary to non-volatile memory. If AltitudeSensor capability is supported, the command is rejected, and the ALS subunit returns DataReadOnly return code.

Message format:

```
PrimaryCommand ALSSetAltitudeCommand {
      CommandCode_t         Command ← 0x0403
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      meter_t               Altitude
      Provenance_t          AltitudeProvenance
      meter_t               AltitudePrecision
      Provenance_t          AltitudePrecisionProvenance
}
```

```
ALDResponse ALSSetAltitudeResponse {
      CommandCode_t         Command ← 0x0403
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t {
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      DataReadOnly
      OutOfRange
      InvalidProvenance
      GeneralError
      OK
}
```

Primary pseudocode:

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
      UNLESS ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = ReadWrite THEN
            RETURN NotAuthorised
            EXIT
      ENDIF
ENDIF
result ← IsCommandAllowed( LIST{      OperatingConnectionState,
                                      RestrictedConnectionState},
                                      Cmd.Command, CurrentPort)
UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF
```

```
IF ALSCapabilities.AltitudeSensor THEN
        RETURN DataReadOnly
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
ENDIF

IF Cmd.Altitude > 10000 meter_t THEN
        RETURN OutOfRange
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
ENDIF
IF Cmd.AltitudePrecision > 100 meter_t THEN
        RETURN OutOfRange
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
ENDIF

IF Cmd.AltitudeProvenance NOT Manual THEN
        RETURN InvalidProvenance
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
ENDIF

IF Cmd.AltitudePrecisionProvenance NOT Manual THEN
        RETURN InvalidProvenance
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
ENDIF

«store the altitude information and its provenance to non-volatile memory»

IF «the ALD detects a hardware error» THEN
        // Replace “Hardware error” with descriptive text to be read using
        // GetDiagnosticInformation
        RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
        RETURN “GeneralError”
ELSE
        RETURN OK
ENDIF
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.12.5. ALS Get Altitude

Description (Informative):

On receipt of this command the ALS subunit shall return the measured altitude information from the on-board sensor, its provenances and precision. If AltitudeSensor capability is not supported, the information set by ALS Set Altitude command is returned.

Message format:

```
PrimaryCommand ALSGetAltitudeCommand {
      CommandCode_t         Command ← 0x0404
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse ALSGetAltitudeResponse {
      CommandCode_t         Command ← 0x0404
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            BOOLEAN         AltitudeProvidedBySensor
            meter_t         Altitude
            Provenance_t    AltitudeProvenance
            meter_t         AltitudePrecision
            Provenance_t    AltitudePrecisionProvenance
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t{
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      OK
      ALSSensorDataNotAvailable
}
```

Primary pseudocode):

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
      IF ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
            RETURN NotAuthorised
            EXIT
      ENDIF
```

```
ENDIF

result ← IsCommandAllowed( LIST{      OperatingConnectionState,
                                      RestrictedConnectionState},
                                      Cmd.Command, CurrentPort)

UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF

// Altitude sensor capability supported, data coming from the on-board sensor

IF ALSCapabilities.AltitudeSensor THEN
      AltitudeProvidedBySensor ← TRUE
      AltitudeProvenance ← Automatic
      AltitudePrecisionProvenance ← Automatic
ELSE
      AltitudeProvidedBySensor ← FALSE
ENDIF

IF ALSCapabilities.AltitudeSensor THEN              // Data from the sensor
      IF «Altitude sensor data available» THEN
            RETURN OK, AltitudeProvidedBySensor, «Measured altitude, altitude precision
            and corresponding provenance from the sensor»
      ELSE
            RETURN ALSSensorDataNotAvailable
            CommandExit(Cmd.Command, CurrentPort)
            EXIT
      ENDIF
ELSE                                                // Data from the memory
      RETURN OK, AltitudeProvidedBySensor, «Altitude, altitude precision, and
      corresponding provenance previously stored by ALSSetAltitude»
ENDIF       // If the ALS was shipped from factory with no data in these fields, the
            // corresponding provenances are NotSet

CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.12.6. ALS Set Orientation Alarm Limits

Description (Informative):

On receipt of this command the ALS subunit shall store the orientation alarm limit information received from primary to non-volatile memory. If OrientationSensor capability is not supported, the command is rejected, and the ALS subunit returns UnsupportedCapability return code.

The alarm limit values set with this command determine at which point orientation alarms are raised.

NOTE: The parameter value check ends when the first failed parameter is found.

Message format:

```
Bitfield DataToBeStored_t : uint8_t {
      MechanicalTiltAlarmLimits       Bit 0      // The upper lower alarm limits values
      MechanicalRollAlarmLimits       Bit 1      // of the same orientation shall be
      MechanicalAzimuthAlarmLimits    Bit 2      // stored as simultaneously
      }

PrimaryCommand ALSSetOrientationAlarmLimitsCommand {
      CommandCode_t         Command ← 0x0405
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      DataToBeStored_t      OrientationAlarmLimitsToBeStored
      degree_t              LowerMechanicalTiltAlarmLimit
      Provenance_t          LowerMechanicalTiltAlarmLimitProvenance
      degree_t              UpperMechanicalTiltAlarmLimit
      Provenance_t          UpperMechanicalTiltAlarmLimitProvenance
      degree_t              LowerMechanicalRollAlarmLimit
      Provenance_t          LowerMechanicalRollAlarmLimitProvenance
      degree_t              UpperMechanicalRollAlarmLimit
      Provenance_t          UpperMechanicalRollAlarmLimitProvenance
      degree_t              MechanicalAzimuthAlarmLimit1
      Provenance_t          MechanicalAzimuthAlarmLimit1Provenance
      degree_t              MechanicalAzimuthAlarmLimit2
      Provenance_t          MechanicalAzimuthAlarmLimit2Provenance
}

ALDResponse ALSSetOrientationAlarmLimitsResponse {
      CommandCode_t         Command ← 0x0405
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            uint8_t               FailedParameterNumber
      }
}

Enumeration ReturnCode_t {
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      UnsupportedCapability
      OutOfRange
      InvalidProvenance
      GeneralError
      OK
}
```

Primary pseudocode:

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
    UNLESS ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = ReadWrite THEN
        RETURN NotAuthorised
        EXIT
    ENDIF
ENDIF

UNLESS ALSCapabilities.OrientationSensor THEN
    Response.FailedParameterNumber ← 1
    RETURN UnsupportedCapability
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF (Cmd.OrientationAlarmLimitsToBeStored.MechanicalTiltAlarmLimits) THEN
    IF Cmd.LowerMechanicalTiltAlarmLimit > 90 degree_t OR
    Cmd.LowerMechanicalTiltAlarmLimit < −90 degree_t OR
    Cmd.UpperMechanicalTiltAlarmLimit > 90 degree_t OR
    Cmd.UpperMechanicalTiltAlarmLimit < −90 degree_t OR
    Cmd.LowerMechanicalTiltAlarmLimit ≥ Cmd.UpperMechanicalTiltAlarmLimit THEN
        Response.FailedParameterNumber ← 2
        RETURN OutOfRange
        EXIT
    ENDIF
    IF Cmd.LowerMechanicalTiltAlarmLimitProvenance NOT Manual OR
    Cmd.UpperMechanicalTiltAlarmLimitProvenance NOT Manual THEN
        Response.FailedParameterNumber ← 2
        RETURN InvalidProvenance
        EXIT
    ENDIF
ENDIF

IF (Cmd.OrientationAlarmLimitsToBeStored.MechanicalRollAlarmLimits) THEN
    IF Cmd.LowerMechanicalRollAlarmLimit > 180 degree_t OR
    Cmd.LowerMechanicalRollAlarmLimit < −180 degree_t OR
    Cmd.UpperMechanicalRollAlarmLimit > 180 degree_t OR
    Cmd.UpperMechanicalRollAlarmLimit < −180 degree_t OR
    Cmd.LowerMechanicalRollAlarmLimit ≥ Cmd.UpperMechanicalRollAlarmLimit THEN
        Response.FailedParameterNumber ← 3
        RETURN OutOfRange
        EXIT
    ENDIF
    IF Cmd.LowerMechanicalRollAlarmLimitProvenance NOT Manual OR
    Cmd.UpperMechanicalRollAlarmLimitProvenance NOT Manual THEN
        Response.FailedParameterNumber ← 3
        RETURN InvalidProvenance
```

```
            EXIT
        ENDIF
ENDIF

IF (Cmd.OrientationAlarmLimitsToBeStored.MechanicalAzimuthAlarmLimits) THEN

        IF Cmd.MechanicalAzimuthAlarmLimit1 > 360 degree_t OR
        Cmd.MechanicalAzimuthAlarmLimit1 < 0 degree_t OR
        Cmd.MechanicalAzimuthAlarmLimit2 > 360 degree_t OR
        Cmd.MechanicalAzimuthAlarmLimit2 < 0 degree_t OR
        Cmd.MechanicalAzimuthAlarmLimit1 = Cmd.MechanicalAzimuthAlarmLimit2 THEN
            Response.FailedParameterNumber ← 4
            RETURN OutOfRange
            EXIT
        ENDIF

        IF Cmd.MechanicalAzimuthAlarmLimit1Provenance NOT Manual OR
        Cmd.MechanicalAzimuthAlarmLimit2Provenance NOT Manual THEN
            Response.FailedParameterNumber ← 4
            RETURN InvalidProvenance
            EXIT
        ENDIF
ENDIF

result ← IsCommandAllowed( LIST{    OperatingConnectionState,
                                    RestrictedConnectionState},
                                    Cmd.Command, CurrentPort)

UNLESS result.allowed THEN
        RETURN result.code
        EXIT
ENDIF

IF (Cmd.OrientationAlarmLimitsToBeStored.MechanicalTiltAlarmLimits) THEN
        «store the lower mechanical tilt alarm limit to non-volatile memory»
        «store the lower mechanical tilt alarm limit provenance to non-volatile memory»
        «store the upper mechanical tilt alarm limit to non-volatile memory»
        «store the upper mechanical tilt alarm limit provenance to non-volatile memory»
ENDIF

IF (Cmd.OrientationAlarmLimitsToBeStored.MechanicalRollAlarmLimits) THEN
        «store the lower mechanical roll alarm limit to non-volatile memory»
        «store the lower mechanical roll alarm limit provenance to non-volatile memory»
        «store the upper mechanical roll alarm limit to non-volatile memory»
        «store the upper mechanical roll alarm limit provenance to non-volatile memory»
ENDIF

IF (Cmd.OrientationAlarmLimitsToBeStored.MechanicalAzimuthAlarmLimits) THEN
        «store the lower mechanical azimuth alarm limit1 to non-volatile memory»
        «store the lower mechanical azimuth alarm limit1 provenance to non-volatile memory»
        «store the upper mechanical azimuth alarm limit2 to non-volatile memory»
        «store the upper mechanical azimuth alarm limit2 provenance to non-volatile memory»
ENDIF
```

IF «the ALD detects a hardware error» THEN
    // Replace “Hardware error” with descriptive text to be read using
    // GetDiagnosticInformation
    RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
    RETURN “GeneralError”
ELSE
    RETURN OK
ENDIF
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.12.7. ALS Get Orientation Alarm Limits

Description (Informative):

On receipt of this command the ALS subunit shall return the orientation alarm limits. If OrientationSensor capability is not supported, the command is rejected, and the ALS subunit returns UnsupportedCapability return code.

Message format:

```
PrimaryCommand ALSGetOrientationAlarmLimitsCommand {
     CommandCode_t        Command ← 0x0406
     CommandSequence_t    PrimaryCommandSequence
     Subunit_t            Subunit
     DataLength_t         DataLength ← 0
}

ALDResponse ALSGetOrientationAlarmLimitsResponse {
     CommandCode_t        Command ← 0x0406
     CommandSequence_t    PrimaryCommandSequence
     ReturnCode_t         ReturnCode
     DataLength_t         DataLength
     if (ReturnCode == OK) {
          degree_t             LowerMechanicalTiltAlarmLimit
          Provenance_t         LowerMechanicalTiltAlarmLimitProvenance
          degree_t             UpperMechanicalTiltAlarmLimit
          Provenance_t         UpperMechanicalTiltAlarmLimitProvenance
          degree_t             LowerMechanicalRollAlarmLimit
          Provenance_t         LowerMechanicalRollAlarmLimitProvenance
          degree_t             UpperMechanicalRollAlarmLimit
          Provenance_t         UpperMechanicalRollAlarmLimitProvenance
          degree_t             MechanicalAzimuthAlarmLimit1
          Provenance_t         MechanicalAzimuthAlarmLimit1Provenance
          degree_t             MechanicalAzimuthAlarmLimit2
          Provenance_t         MechanicalAzimuthAlarmLimit2Provenance
     }
     else {

          ALDState_t           ALDState
          ConnectionState_t    ConnectionState
     }
}
```

```
Enumeration ReturnCode_t{
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      UnsupportedCapability
      OK
}
```

Primary pseudocode):

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
      IF ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
            RETURN NotAuthorised
            EXIT
      ENDIF
ENDIF
result ← IsCommandAllowed( LIST{      OperatingConnectionState,
                                      RestrictedConnectionState},
                                      Cmd.Command, CurrentPort)
UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF
UNLESS ALSCapabilities.OrientationSensor THEN
      RETURN UnsupportedCapability
      CommandExit(Cmd.Command, CurrentPort)
      EXIT
ENDIF
RETURN OK, «Orientation alarm limits and their corresponding provenances»
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.12.8. ALS Set Orientation

Description (Informative):

On receipt of this command the ALS subunit shall store the orientation information and related precision received from primary to non-volatile memory. If OrientationSensor capability is supported, the command is rejected, and the ALS subunit returns DataReadOnly return code.

Message format:

```
PrimaryCommand ALSSetOrientationCommand {
      CommandCode_t         Command ← 0x0413
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      degree_t              MechanicalTilt
      Provenance_t          MechanicalTiltProvenance
      degree_t              MechanicalRoll
      Provenance_t          MechanicalRollProvenance
      degree_t              MechanicalAzimuth
      Provenance_t          MechanicalAzimuthProvenance
}

ALDResponse ALSSetOrientationResponse {
      CommandCode_t         Command ← 0x0403
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t {
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      DataReadOnly
      OutOfRange
      InvalidProvenance
      GeneralError
      OK
}
```

Primary pseudocode):

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
        UNLESS ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = ReadWrite THEN
                RETURN NotAuthorised
                EXIT
        ENDIF
ENDIF

result ← IsCommandAllowed( LIST{        OperatingConnectionState,
                                        RestrictedConnectionState},
                                        Cmd.Command, CurrentPort)

UNLESS result.allowed THEN
        RETURN result.code
        EXIT
ENDIF
```

```
IF ALSCapabilities.OrientationSensor THEN
    RETURN DataReadOnly
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.MechanicalTilt < −180 degree_t OR Cmd.MechanicalTilt > 180 degree_t THEN
    RETURN OutOfRange
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.MechanicalTiltProvenance NOT Manual THEN
    RETURN InvalidProvenance
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.MechanicalRoll < −180 degree_t OR Cmd.MechanicalRoll > 180 degree_t THEN
    RETURN OutOfRange
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.MechanicalRollProvenance NOT Manual THEN
    RETURN InvalidProvenance
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.MechanicalAzimuth < 0 degree_t OR
    Cmd.MechanicalAzimuth > 360 degree_t THEN
    RETURN OutOfRange
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF Cmd.MechanicalAzimuthProvenance NOT Manual THEN
    RETURN InvalidProvenance
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

«store the orientation information and the related provenances to non-volatile memory»

IF «the ALD detects a hardware error» THEN
    // Replace “Hardware error” with descriptive text to be read using
    // GetDiagnosticInformation
    RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
    RETURN “GeneralError”
ELSE
    RETURN OK
ENDIF
```

CommandExit(Cmd.Command, CurrentPort)
EXIT

11.12.9. ALS Get Orientation

Description (Informative):

On receipt of this command the ALS subunit shall return measured orientation information from the on-board sensor and its provenances. If OrientationSensor capability is not supported, the information set by ALS Set Orientation command is returned.

Message format:

```
PrimaryCommand ALSGetOrientationCommand {
      CommandCode_t         Command ← 0x0409
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse ALSGetOrientationResponse {
      CommandCode_t         Command ← 0x0409
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            BOOLEAN               OrientationProvidedBySensor
            degree_t              MechanicalTilt
            Provenance_t          MechanicalTiltProvenance
            degree_t              MechanicalRoll
            Provenance_t          MechanicalRollProvenance
            degree_t              MechanicalAzimuth
            Provenance_t          MechanicalAzimuthProvenance
      }
      else {

            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t{
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      OK
      ALSSensorDataNotAvailable
}
```

Primary pseudocode):

(This section is intentionally left blank)

ALD pseudocode:

```
BOOLEAN OrientationDataValid

IF ALDType = MALD THEN
        IF ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
                RETURN NotAuthorised
                EXIT
        ENDIF
ENDIF

result ← IsCommandAllowed( LIST{        OperatingConnectionState,
                                        RestrictedConnectionState},
                                        Cmd.Command, CurrentPort)

UNLESS result.allowed THEN
        RETURN result.code
        EXIT
ENDIF

// Altitude orientation sensor capability supported, data coming from the on-board sensor
IF ALSCapabilities.OrientationSensor THEN
        OrientationProvidedBySensor ← TRUE
        MechanicalTiltProvenance ← Automatic
        MechanicalRollProvenance ← Automatic
        MechanicalAzimuthProvenance ← Automatic
ELSE
        OrientationProvidedBySensor ← FALSE
ENDIF

IF ALSCapabilities.OrientationSensor THEN           // Data from the sensor
        IF «Orientation sensor data available» THEN
                RETURN OK, LocationProvidedBySensor, «Measured orientation information
                and corresponding provenances from the sensor»
        ELSE
                RETURN ALSSensorDataNotAvailable
                CommandExit(Cmd.Command, CurrentPort)
                EXIT
        ENDIF
ELSE                                                // Data from memory
        RETURN OK, LocationProvidedBySensor, «Orientation information and
        corresponding provenances previously stored by ALSSetOrientation»
ENDIF       // If the ALS was shipped from factory with no data  in these fields, the
                // corresponding provenances are NotSet

CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.12.10. ALS Get Monitored Orientation Information

Description (Informative):

On receipt of this command the ALS subunit shall return the minimum and maximum monitored orientation information since the last time the values were cleared. This information may be used to analyse the effect of sway or other mechanical disturbances. If orientation monitoring has not yet recorded any value for a monitored parameter, the value +3276.7 degree_t is returned to indicated this.

The monitored values shall be cleared during ALD reset or when this command is executed with the flag ClearMonitoredOrientationValues ← TRUE.

If OrientationSensor capability is not supported, the command is rejected, and the ALS subunit returns UnsupportedCapability return code.

Message format:

```
PrimaryCommand ALSGetMonitoredOrientationInformationCommand {
      CommandCode_t         Command ← 0x0410
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 1
      BOOLEAN               ClearMonitoredOrientationValues
}

ALDResponse ALSGetMonitoredOrientationInformationResponse {
      CommandCode_t         Command ← 0x0410
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      degree_t              LowestMonitoredMechanicalTilt
      Provenance_t          LowestMonitoredMechanicalTiltProvenance
      degree_t              HighestMonitoredMechanicalTilt
      Provenance_t          HighestMonitoredMechanicalTiltProvenance
      degree_t              LowestMonitoredMechanicalRoll
      Provenance_t          LowestMonitoredMechanicalRollProvenance
      degree_t              HighestMonitoredMechanicalRoll
      Provenance_t          HighestMonitoredMechanicalRollProvenance
      degree_t              LowestMonitoredMechanicalAzimuth
      Provenance_t          LowestMonitoredMechanicalAzimuthProvenance
      degree_t              HighestMonitoredMechanicalAzimuth
      Provenance_t          HighestMonitoredMechanicalAzimuthProvenance
      }
      else {

            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}
```

```
Enumeration ReturnCode_t{
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      UnsupportedCapability
      OK
}
```

Primary pseudocode):

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
      IF ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
            RETURN NotAuthorised
            EXIT
      ENDIF
ENDIF
result ← IsCommandAllowed( LIST{      OperatingConnectionState,
                                      RestrictedConnectionState},
                                      Cmd.Command, CurrentPort)
UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF
```

```
UNLESS ALSCapabilities.OrientationSensor THEN
      RETURN UnsupportedCapability
      CommandExit(Cmd.Command, CurrentPort)
      EXIT
ENDIF

// Monitored data provenances are automatic because they source is always the sensor
LowestMonitoredMechanicalTiltProvenance ← Automatic
HighestMonitoredMechanicalTiltProvenance ← Automatic
LowestMonitoredMechanicalRollProvenance ← Automatic
HighestMonitoredMechanicalRollProvenance ← Automatic
LowestMonitoredMechanicalAzimuthProvenance ← Automatic
HighestMonitoredMechanicalAzimuthProvenance ← Automatic

RETURN OK, «Lowest and highest orientation values and their corresponding provenances»

IF ClearMonitoredOrientationValues THEN
      LowestMonitoredMechanicalTilt ← +3276.7          // Indicates a NULL value
      HighestMonitoredMechanicalTilt ← +3276.7         // That is, orientation monitoring
      LowestMonitoredMechanicalRoll ← +3276.7          // has not yet recorded any value
      HighestMonitoredMechanicalRoll ← +3276.7         // for that monitored parameter
      LowestMonitoredMechanicalAzimuth ← +3276.7       // after they were cleared
      HighestMonitoredMechanicalAzimuth ← +3276.7
ENDIF

CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.12.11. ALS Set Array Element Numbers to Subunit

Description (Informative):

On receipt of this command the ALS subunit shall store the provided list of array element numbers to which the ALS subunit has a logical relationship.

The ALD vendor may have defined one or more array element numbers as read only. If the array element numbers are not stored, the ALD will return the DataReadOnly return code.

Message format:

```
PrimaryCommand ALSSetArrayElementNumbersToSubunitCommand {
      CommandCode_t         Command ← 0x0411
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      uint8_t               NrOfArrayElements
      uint16_t              ArrayElementNumbers[1..NrOfArrayElements]
      Provenance_t          ArrayElementNumbersProvenance
}
```

```
ALDResponse ALSSetArrayElementNumbersToSubunitResponse {
      CommandCode_t         Command ← 0x0411
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t {
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      InvalidArrayElementNumber
      DataReadOnly
      GeneralError
      OK
}
```

Primary pseudocode:

(This section is intentionally left blank)

ALD pseudocode:

```
IF ALDType = MALD THEN
      UNLESS ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = ReadWrite THEN
            RETURN NotAuthorised
            EXIT
      ENDIF
ENDIF
result ← IsCommandAllowed( LIST{      OperatingConnectionState,
                                      RestrictedConnectionState},
                                      Cmd.Command, CurrentPort)
UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF
```

UNLESS Cmd.ArrayElementNumber IN ArrayElementNumberList THEN
RETURN InvalidArrayElementNumber
CommandExit(Cmd.Command, CurrentPort)
EXIT
ENDIF

IF «Array element numbers are read only in this ALS subunit» THEN
RETURN DataReadOnly
ELSE
«Store the array element numbers and their provenance for the supplied Subunit to non-volatile memory»

IF «the ALD detects a hardware error» THEN
// Replace “Hardware error” with descriptive text to be read using
// GetDiagnosticInformation
RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
RETURN “GeneralError”
ELSE
RETURN OK
ENDIF
ENDIF
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.12.12. ALS Get Array Element Numbers Of Subunit

Description (Informative):

On receipt of this command the ALS subunit shall return the list of array elements to which the ALS subunit has a logical relationship.

Message format:

```
PrimaryCommand ALSGetArrayElementNumbersOfSubunitCommand {
      CommandCode_t         Command ← 0x0412
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse ALSGetArrayElementNumbersOfSubunitResponse {
      CommandCode_t         Command ← 0x0412
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            uint8_t         NrOfArrayElements
            uint16_t        ArrayElementNumbers[1..NrOfArrayElements]
            Provenance_t    ArrayElementNumbersProvenance
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}
```

```
Enumeration ReturnCode_t{
      // The following are return codes from command message validation (see section
      // 12.6.2 in [1]
      FormatError
      UnknownCommand
      InvalidSubunitNumber
      ProtocolVersionNotNegotiated
      InvalidSubunitType
      // The following are return codes from command pseudocode below
      NotAuthorised
      NotAControlPort
      IncorrectState
      Busy
      InUseByAnotherPrimary
      OK
}
```

Primary pseudocode):

(This section is intentionally left blank)

ALD pseudocode:

IF ALDType = MALD
AND ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
RETURN NotAuthorised
EXIT
ENDIF

result ← IsCommandAllowed(LIST{ OperatingConnectionState,
RestrictedConnectionState,
MALDSetupConnectionState},
Cmd.Command, CurrentPort)

UNLESS result.allowed THEN
RETURN result.code
EXIT
ENDIF
RETURN OK, «for the requested ALS subunit, return the number of stored array elements, the list of array numbers, and their provenance »

CommandExit(Cmd.Command, CurrentPort)

EXIT

ANNEX A: Mechanical Azimuth, Tilt and Roll data representation (Informative):

This annex shows how the real world mechanical azimuth, tilt and roll data is encoded.

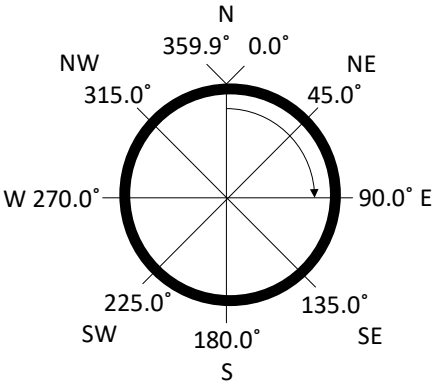

Figure A-1: Mechanical Azimuth data representation

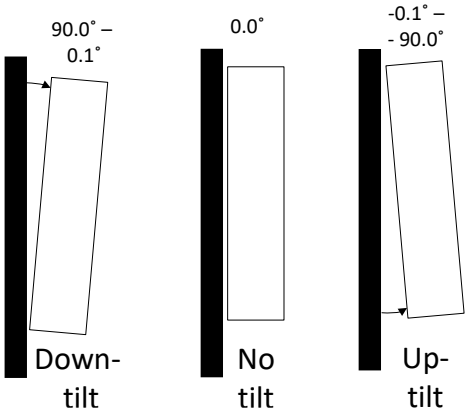

Figure A-2: Mechanical Tilt data representation

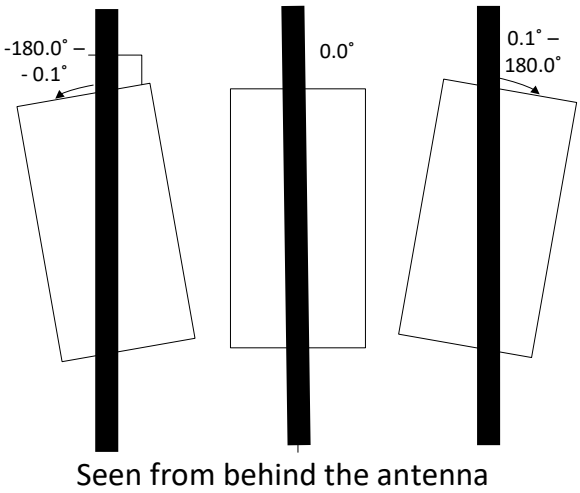

Figure A-3: Mechanical Roll data representation

ANNEX B: Mechanical alarm limits (Informative):

This annex clarifies the functioning of the tilt, roll and azimuth-alarm limits.

B-1.1 Mechanical tilt alarm limits

In the following tilt alarm limits example the following values are used:

- LowerMechanicalTiltAlarmLimit ← 7.0°
- UpperMechanicalTiltAlarmLimit ← 13.0°

NOTE: LowerMechanicalTiltAlarmLimit shall always be lower than the UpperMechanicalTiltAlarmLimit, and both shall not be the same!

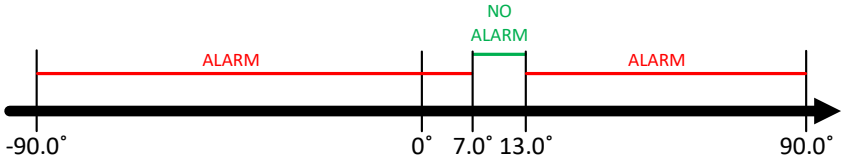

Figure B-1.1-1: Tilt alarm limits

B-1.2 Mechanical roll alarm limits

In the following roll alarm limits example the following values are used:

- LowerMechanicalRollAlarmLimit ← −5.0°
- UpperMechanicalRollAlarmLimit ← 5.0°

NOTE: LowerMechanicalRollAlarmLimit shall always be lower than the UpperMechanicalRollAlarmLimit, and both shall not be the same!

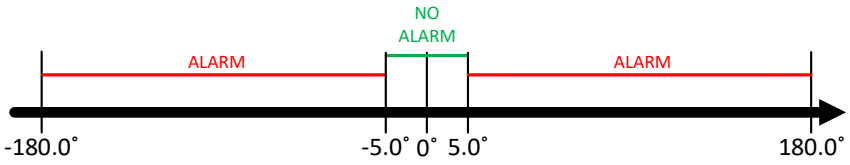

Figure B-1.2-1: Roll alarm limits

B-1.3 Mechanical azimuth alarm limits

When defining alarms on a compass circle, it is not enough to just define two alarm limits. This is because just defining two alarm limits divides the compass circle into two sectors, but leaves unclear in which sector there will be an alarm.

The solution adopted in this standard is to not to specify a required order for the two limits. The azimuth alarm limits are called Limit1 and Limit2.

The examples in sections B-1.3.1. and B-1.3.2. shows the alarm functionality in these cases.

The section B-1.3.3. explains the logic that defines in which part of the compass circle the alarm is off and in which part it is on.

B-1.3.1 MechanicalAzimuthAlarmLimit1 > MechanicalAzimuthAlarmLimit2

In the following tilt alarm limits example the following values are used:

- MechanicalAzimuthAlarmLimit1 ← 315.0°
- MechanicalAzimuthAlarmLimit2 ← 45.0°

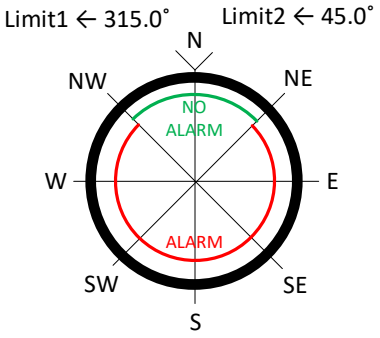

Figure B-1.3.1-1: Azimuth alarm limits, Limit1 > Limit2

B-1.3.2 MechanicalAzimuthAlarmLimit1 < MechanicalAzimuthAlarmLimit2

In the following tilt alarm limits example the following values are used:

- MechanicalAzimuthAlarmLimit1 ← 45.0°
- MechanicalAzimuthAlarmLimit2 ← 315.0°

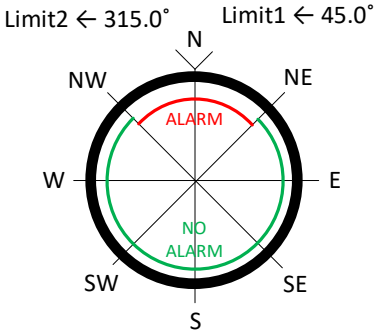

Figure B-1.3.2-1: Azimuth alarm limits, Limit1 < Limit2

B-1.3.3 Mechanical azimuth alarm logic

The following logic determines the alarm status:

IF MechanicalAzimuthAlarmLimit1 > MechanicalAzimuthAlarmLimit2 THEN
    ON «Measured mechanical azimuth» < MechanicalAzimuthAlarmLimit1 AND
    «Measured mechanical azimuth» > MechanicalAzimuthAlarmLimit2
        RAISE ALSAlarmMechanicalAzimuth SEVERITY Major ON Subunit
    DONE
ELSE
    ON «Measured mechanical azimuth» < MechanicalAzimuthAlarmLimit1 OR
    «Measured mechanical azimuth» > MechanicalAzimuthAlarmLimit2
        RAISE ALSAlarmMechanicalAzimuth SEVERITY Major ON Subunit
    DONE
ENDIF

NOTE: MechanicalAzimuthAlarmLimit1 and MechanicalAzimuthAlarmLimit2 shall not be the same!