Antenna Database
AISG-ST-ADB
vADB3.1.2.1

Revision History

DATE	ISSUE	NOTES
27th May 2019	vADB3.1.2.1	Second public release
5th November 2018	vADB3.1.1.6	First public release

1. FOREWORD

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 covers the antenna line devices capable of storing the antenna information.

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.

This standard is independent of previous 3GPP specifications.

2. SCOPE

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 Database (ADB) subunit.

This standard defines the functional behaviour of ADB subunits. The text of the standard defines explicitly what is required or permitted. Anything that is not explicitly allowed is not permitted.

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	AISG APCC: “Antenna port colour coding standard”
4	170217 NGMN P-BASTA Whitepaper v10.0: “Recommendation on Base Station Antenna Standards”

4. VERSION COMPLIANCE

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

5. ABBREVIATIONS

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

ADB	Antenna Database
ALD	Antenna Line Device
ID	Identifier
LHC	Left Hand Circular (Polarisation)
MALD	Multi-primary ALD
NGMN	Next Generation Mobile Networks
RET	Remote Electrical Tilt
RF	Radio Frequency
RHC	Right Hand Circular (Polarisation)
SALD	Single-primary ALD
TCC	Time-Consuming Command
Xpol+	Cross Polarisation, slant +45 degrees
Xpol-	Cross Polarisation, slant -45 degrees
3GPP	3rd Generation Partnership Project

6. TERMINOLOGY

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

Antenna port	RF port of the antenna with direction towards the basestation.
Array observation direction reference	The direction from which the antenna is viewed to define its polarisation. Possible values are front and back. (Some vendors determine the polarisation by looking at the antenna from the front and others by looking at the antenna from the back, see Annex A).
Array polarisation sense reference	The sense in which the 45 degrees polarisation angle is specified. Vertical axis is used as starting point. Possible values are clockwise or counter-clockwise. This assumes that the antenna is installed in its intended orientation.
Mechanical bearing	The direction orthogonal to the axis of the antenna assembly, expressed in degrees East of True North (ETN).
Mechanical tilt	Tilt angle of the antenna in the 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 a negative number. Tilt is reported in decimal degrees, to one decimal place of accuracy, and then multiplied by 10 so that is may be represented by an integer.
Polarisation	Orientation of the electric field vector of the radio wave emitted by a logical array of an antenna.

7. DEFINITIONS

```
CONSTANT uint16_t NrOfSubunitAlarms ← 0// Number of subunit alarm types for this
subunit type
```

8. GENERAL ASPECTS

8.1. General

An Antenna DataBase (ADB) subunit stores data about the antenna, its contained arrays, installation and site mapping. Any antenna enclosure shall contain one and only one ADB subunit.

It is not allowed to implement an ADB subunit outside an antenna enclosure.

8.2. Subunit relationship

An ADB subunit has logical relationship with all ports of the ALD having antenna.

8.3. Logical array and array ID

A logical array is a basic array element of an antenna, defined in [1]. Logical arrays are differentiated by a logical array number. This numbering scheme is vendor specific.

Array ID, defined in [3], may contain several logical arrays, and it is not used in this subunit type standard. It is used in a label close to an antenna port to help installers to cable the site correctly.

8.4. Return codes

This subunit type standard introduces the following subunit type specific return codes.

```
Enumeration ReturnCode_t : uint16_t {
     ADBNotAntennaPort ← 0x0300                // e.g. a RET-port
}
```

8.5. Resumption of operation

The following data shall be retained after reset:

- Antenna installation data
- RFPathID to logical array data

9. LAYER 1

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

9.1. DC power consumption

This subunit type standard does not define the power consumption of an ADB. ADB is not able to switch the ALD to HighPowerMode.

10. LAYER 2

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

11. LAYER 7

An ALD which contains an ADB subunit 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
ADB	0x03

Table 11.1-1: Subunit type code

11.2. Overview of commands for ADB subunits

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

The following abbreviations are used in the Table 11.2-1: “Command set for ADB subunits”.

M Mandatory

O Optional

- Not applicable

	Code	Initiator	Subunit	Timeout	TCC	Mandatory for:			Changes the ConnectionState	Changes to HighPowerMode	Minimum required authority
						Primary	SALD	MALD			
ADB commands											
ADB Get Antenna Info	0x0300	Primary	>0	1 s	no	M	M	M	no	no	RO
ADB Get Antenna Port Info	0x0301	Primary	>0	1 s	no	M	M	M	no	no	RO
ADB Get Antenna Logical Array Info	0x0302	Primary	>0	1 s	no	M	M	M	no	no	RO
ADB Set Antenna Installation Info	0x0303	Primary	>0	1 s	no	M	M	M	no	no	RW
ADB Get Antenna Installation Info	0x0304	Primary	>0	1 s	no	M	M	M	no	no	RO
Site mapping commands for ADB											
ADB Set RF Path ID to Logical Array	0x0305	Primary	>0	1 s	no	O	M	M	no	no	RW
ADB Get RF Path ID of Logical Array	0x0306	Primary	>0	1 s	no	O	M	M	no	no	RO

Table 11.2-1: Command set for ADB subunits

11.3. Bearing representation

The bearing range supported is from 0.0° to +359.9° East of True North. The bearing value is expressed in 0.1° units with a range of 0 to +3599.

11.4. Mechanical tilt representation

The mechanical tilt range supported is −90.0 – +90.0. The mechanical tilt is expressed in 0.1° units with a range of −900 – +900.

11.5 Polarisation representation

Vendors define the polarisation value of an array as either negative and positive slant linear polarisation. [Value: Xpol+ or Xpol−]. This way of defining polarisation is not comparable between different vendors due to the differences in the polarisation references they use.

This standard defines a way to compose a vendor independent and compatible slant polarisation description by introducing two references that indicate the way that a vendor is defining polarisation. By combining the polarisation value and these two references, i.e. array observation direction reference and array polarisation sense reference, all specified by the manufacturer during ALD production and available through the ADB commands, vendor independent and comparable cross polarisation information can be derived. This method is only used for slant linear polarisation.

Array observation direction reference: Array polarisation is defined looking at the antenna from the from or from the back [Value: Front or Back].

Array polarisation sense reference: Array slant polarisation is defined referring to vertical. +45 degree polarisation may be defined as being oriented 45 degrees clockwise from vertical or 45 degrees counter-clockwise from vertical. [Value: CW or CCW].

Conversion from vendor specific polarisation to comparable polarisation is done using the table 11.5-1.

Polarisation Value	Array Observation Direction Reference	Array Polarisation Sense Reference	Comparable Polarisation ID
Xpol+	Front	CW	A
Xpol−	Front	CW	B
Xpol+	Back	CW	B
Xpol−	Back	CW	A
Xpol+	Front	CCW	B
Xpol−	Front	CCW	A
Xpol+	Back	CCW	A
Xpol−	Back	CCW	B

Table 11.5-1: Polarisation conversion

For examples of this method see Annex A.

11.6. ADB commands

11.6.1. ADB Get Antenna Info

Description (Informative):

This command returns antenna information.

The behaviour of the antenna model number and antenna serial number fields are vendor specific and it is possible that if a field replaceable submodule is exchanged in the field, info in these fields changes or is lost. The user may have to maintain this information manually during a field submodule replacement as it is not mandatory for the vendor to provide a mechanism for reading this information to the new submodule from the antenna.

Message format (Normative):

```
PrimaryCommand ADBGetAntennaInfoCommand {
      CommandCode_t         Command ← 0x0300
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse ADBGetAntennaInfoResponse {
      CommandCode_t         Command ← 0x0300
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            uint8_t         LengthOfAntennaModelNumber      // max 32 octets
            UTF8String_t    AntennaModelNumber
            Provenance_t    AntModelNumberProvenance
            uint8_t         LengthOfAntennaSerialNumber     // max 32 octets
            UTF8String_t    AntennaSerialNumber
            Provenance_t    AntSerialNumberProvenance
            uint8_t         NrOfLogicalArrays
            Provenance_t    NrOfLogicalArraysProvenance
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState

      }
}

Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorized
      IncorrectState
}
```

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

```
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, «AntennaModelNumber, AntennaSerialNumber, NrOfLogicalArrays and the
corresponding lengths and provenances»
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.6.2. ADB Get Antenna Port Info

Description (Informative):

This command returns antenna port information.

Message format (Normative):

```
PrimaryCommand ADBGetAntennaPortInfoCommand {
      CommandCode_t         Command ← 0x0301
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 2
      uint16_t              PortNumber
}

ALDResponse ADBGetAntennaPortInfoResponse {
      CommandCode_t         Command ← 0x0301
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            uint8_t    NrOfArraysConnectedToThePort
            uint16_t
      LogicalArraysConnectedToThePort[NrOfLogicalArraysConnectedToThePort]
            Provenance_t    LogicalArraysConnectedToThePortProvenance
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}
```

```
Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorised
      InCorrectState
      InvalidPortNumber
      ADBNotAntennaPort
}
```

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

```
IF Cmd.PortNumber < 1 OR Cmd.PortNumber > MaxPort THEN
      RETURN InvalidPortNumber
      EXIT
ENDIF

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

IF (PortProperties[Cmd.PortNumber] bitwise AND RF) ≠ RF THEN
      RETURN ADBNotAntennaPort
      EXIT
ENDIF

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

UNLESS result.allowed THEN
      RETURN result.code
      EXIT

RETURN OK, «number of logical arrays connected to the port, their array numbers and the
corresponding provenance»
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.6.3. ADB Get Antenna Logical Array Info

Description (Informative):

This command returns the properties of the logical arras contained within an antenna.

Message format (Normative):

```
Bitfield ArrayPolarisation_t : uint8_t {
      Xpol+      : Bit 0
      Xpol-      : Bit 1
      Vertical   : Bit 2
      Horizontal : Bit 3
      RHC        : Bit 4
      LHC        : Bit 5
}

Enumeration ArrayObservationReference_t : uint8_t {
      Front ← 0            // Polarisation is observed by looking at the
                           // antenna from the front for Xpol+ or Xpol-
      Back ← 1             // Polarisation is observed by looking at the
                           // antenna from the back for Xpol+ or Xpol-
      NotApplicable ← 2    // This reference is not applicable to LHC, RHC
                           // Vertical and Horizontal polarisation
}

Enumeration ArrayPolarisationSenseReference_t : uint8_t {
      CW ← 0               // Sense is 45 deg clockwise from vertical
                           // for Xpol+ and Xpol-
      CCW ← 1              // Sense is 45 deg counter-clockwise from vertical
                           // for Xpol+ and Xpol-
      NotApplicable ← 2    // This reference is not applicable to LHC, RHC
                           // Vertical and Horizontal polarisations
}

PrimaryCommand ADBGetAntennaLogicalArrayInfoCommand {
      CommandCode_t        Command ← 0x0302
      CommandSequence_t    PrimaryCommandSequence
      Subunit_t            Subunit
      DataLength_t         DataLength ← 2
      uint16_t             LogicalArrayNumber
}
```

```
ALDResponse ALDGetAntennaLogicalArrayInfoResponse {
      CommandCode_t         Command ← 0x0302
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            uint16_t                      RelativeArrayPositionX     // These
            Provenance_t                  RelativeArrayPositionXProvenance // will
            uint16_t                      RelativeArrayPositionY     // be defined
            Provenance_t                  RelativeArrayPositionYProvenance // later
            uint16_t                      AzimuthBeamwidth3dB  // see [4]
                                                               // paragraph 3.2.5
            Provenance_t                  AzimuthBeamwidth3dBProvenance
            uint16_t                      Gain                 // max value in dBi
            Provenance_t                  GainProvenance
            uint8_t                       NrOfFrequencies
            FrequencyRange_t              ArrayFrequencies[NrOfFrequencies]
                                                               // See Chapter 13 in
                                                               // AISG v3.0 base
                                                               // standard
            Provenance_t                  ArrayFrequenciesProvenance
            ArrayPolarisation_t           ArrayPolarisation[3]
            Provenance_t                  ArrayPolarisationProvenance
            ArrayObservationReference_t   ArrayObservationReference
            Provenance_t                  ArrayObservationReferenceProvenance
            PolarisationSenseReference_t  ArrayPolarisationSenseReference
            Provenance_t                  ArrayPolarisationSenseReferenceProvenance
      }
      else {
            ALDState_t           ALDState
            ConnectionState_t    ConnectionState
      }
}

Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorised
      IncorrectState
      InvalidArrayNumber
}
```

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

```
IF Cmd.LogicalArrayNumber < 1 OR Cmd.ArrayLogicalNumber > MaxLogicalArray THEN
      RETURN InvalidLogicalArrayNumber
      EXIT
ENDIF
```

```
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, «Logical array information and the corresponding provenances»
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.6.4. ADB Set Antenna Installation Info

Description (Informative):

This command stores installation data and corresponding provenances in non-volatile memory.

Only if an automated tool is used to generate the information to be written and automatically transfer it to the primary, the primary shall set the provenance to Automatic. Otherwise the primary shall set the provenance to Manual.

The bitfield InstallationDataToBeStored controls which data fields are stored. The same bit in the bitfield controls the storage of the length of the data (where applicable), the data itself and its provenance as follows:

Bit value 1: Corresponding data is stored together with length (where applicable) and provenance. Existing data is overwritten.

Bit value 0: No data is written to the non-volatile memory and existing data is preserved. The corresponding data in the message is ignored. It is recommended that such data shall be zeroed.

Message format (Normative):

```
Bitfield DataToBeStored_t : uint8_t {
      SectorID             : Bit 0
      PositionWithinSector : Bit 1
      MechanicalBearing    : Bit 2
      MechanicalTilt       : Bit 3
}
```

```
PrimaryCommand ADBSetAntennaInstallationInfoCommand {
      CommandCode_t         Command ← 0x0303
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      DataToBeStored_t      InstallationDataToBeStored
      uint8_t               LengthOfSectorID                // max 32 octet
      TextString_t          SectorID
      Provenance_t          SectorIDProvenance
      uint8_t               LengthOfPositionWithinSector
      TextString_t          PositionWithinSector            // This is vendor specific
      Provenance_t          PositionWithinSectorProvenance
      uint16_t              MechanicalBearing
      Provenance_t          MechanicalBearingProvenance
      uint16_t              MechanicalTilt                  // Mechanical
                                                            // tilt in
                                                            // degrees
      Provenance_t          MechanicalTiltProvenance
}

ALDResponse ADBSetAntennaInstallationInfoResponse {
      CommandCode_t         Command ← 0x0303
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            uint8_t               ParameterNumber
      }
}

Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorised
      IncorrectState
      OutOfRange
      InvalidProvenance
      GeneralError
}
```

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

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

```
result ← IsCommandAllowed( LIST{    OperatingConnectionState,
                                    RestrictedConnectionState},
                                    Cmd.Command, CurrentPort)
UNLESS result.allowed THEN
    RETURN result.code
    EXIT
ENDIF

IF (Cmd.InstallationDataToBeStored.BaseStationID) THEN
    IF Cmd.LengthOfBaseStationID > 32 THEN
        Response.ParameterNumber ← 1
        RETURN OutOfRange
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
    ELSEIF Cmd.BaseStationIDProvenance NOT IN (Manual, Automatic) THEN
        Response.ParameterNumber ← 1
        RETURN InvalidProvenance
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
    ELSE
        Store the LengthOfBasestationID, BaseStationID and BaseStationIDProvenance
        to non-volatile memory
    ENDIF
ENDIF

IF (Cmd.InstallationDataToBeStored.SectorID) THEN
    IF Cmd.LengthOfSectorID > 32 THEN
        Response.ParameterNumber ← 2
        RETURN OutOfRange
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
    ELSEIF Cmd.SectorIDProvenance NOT IN (Manual, Automatic) THEN
        Response.ParameterNumber ← 2
        RETURN InvalidProvenance
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
    ELSE
    Store the LengthOfSectorID, SectorID and SectorIDProvenance to non-volatile memory
    ENDIF
ENDIF

IF (Cmd.InstallationDataToBeStored.LengthOfPositionWithinSector) THEN
    IF Cmd.LengthOfPositionWithinSector > 32 THEN
        Response.ParameterNumber ← 3
        RETURN OutOfRange
        CommandExit(Cmd.Command, CurrentPort)
        EXIT
```

```
        ELSEIF Cmd.PositionWithinSectorProvenance NOT IN (Manual, Automatic) THEN
                Response.ParameterNumber ← 3
                RETURN InvalidProvenance
                CommandExit(Cmd.Command, CurrentPort)
                EXIT
        ELSE
                Store the LengthOfPositionWithinSector, PositionWithinSector and
                PositionWithinSectorProvenance to non-volatile memory
        ENDIF
ENDIF

IF (Cmd.InstallationDataToBeStored.MechanicalBearing) THEN
        IF Cmd.MechanicalBearing > 3599 THEN
                Response.ParameterNumber ← 4
                RETURN OutOfRange
                CommandExit(Cmd.Command, CurrentPort)
                EXIT
        ELSEIF Cmd.MechanicalBearingProvenance NOT IN (Manual, Automatic) THEN
                Response.ParameterNumber ← 4
                RETURN InvalidProvenance
                CommandExit(Cmd.Command, CurrentPort)
                EXIT
        ELSE
                Store the MechanicalBearing and MechanicalBearingProvenance to non-volatile
                memory
        ENDIF
ENDIF

IF (Cmd.InstallationDataToBeStored.MechanicaTilt) THEN
        IF Cmd.MechanicalTilt > 900 OR Cmd.MechanicalTilt < −900 THEN
                Response.ParameterNumber ← 5
                RETURN OutOfRange
                CommandExit(Cmd.Command, CurrentPort)
                EXIT
        ELSEIF Cmd.MechanicalTitProvenance NOT IN (Manual, Automatic) THEN
                Response.ParameterNumber ← 5
                RETURN InvalidProvenance
                CommandExit(Cmd.Command, CurrentPort)
                EXIT
        ELSE
                Store the MechanicalTilt and MechanicalTiltProvenance to non-volatile
                memory
        ENDIF
ENDIF
```

IF the ALD detects a hardware error THEN
RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
RETURN GeneralError
ELSE
RETURN OK
ENDIF

CommandExit(Cmd.Command, CurrentPort)
EXIT

11.6.5. ADB Get Antenna Installation Info

Description (Informative):

This command returns the installation data from the non-volatile memory.

Message format (Normative):

```
PrimaryCommand ADBGetAntennaInstallationInfoCommand {
     CommandCode_t         Command ← 0x0304
     CommandSequence_t     PrimaryCommandSequence
     Subunit_t             Subunit
     DataLength_t          DataLength ← 0
}

ALDResponse ADBGetAntennaInstallationInfoResponse {
     CommandCode_t         Command ← 0x0304
     CommandSequence_t     PrimaryCommandSequence
     ReturnCode_t          ReturnCode
     DataLength_t          DataLength
     if (ReturnCode == OK) {
          uint8_t               LengthOfSectorID               // max 32 octet
          TextString_t          SectorID
          Provenance_t          SectorIDProvenance
          uint8_t               LengthOfPositionWithinSector
          TextString_t          PositionWithinSector
          Provenance_t          PositionWithinSectorProvenance
          uint16_t              MechanicalBearing
          Provenance_t          MechanicalBearingProvenance
          uint16_t              MechanicalTilt            // Mechanical
                                                          // tilt in
                                                          // degrees
          Provenance_t          MechanicalTiltProvenance
     }
     else {
          ALDState_t            ALDState
          ConnectionState_t     ConnectionState
     }
}
```

```
Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      IncorrectState
      InvalidLogicalArrayNumber
      NotAuthorised
}
```

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

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

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

UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF

RETURN OK, «SectorID, PositionWithinSector, MechanicalBearing, MechanicalTilt and the
corresponding lengths and provenances»
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.6.6. ADB Set RF Path ID To Logical Array

Description (Informative):

This command assigns the list of RF Path IDs and provenances, associated with the requesting primary, to the specified logical array.

If an automated tool is used to generate the information to be written and automatically transfer it to the primary, the primary shall set the provenance to Automatic. Otherwise the primary shall set the provenance to Manual.

Message format (Normative):

NOTE: After any antenna line configuration change, the mapping of the RF Path ID must be revalidated and possibly regenerated.

```
PrimaryCommand ADBSetRFPathIDToLogicalArrayCommand {
      CommandCode_t         Command ← 0x0305
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      uint16_t              LogicalArrayNumber
      Provenance_t          Provenance
      uint8_t               NrOfRFPathIDs
      uint16_t              RFPathID[NrOfRFPathIDs]
}

ALDResponse ADBSetRFPathIDToLogicalArrayResponse {
      CommandCode_t         Command ← 0x0305
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
      }
}

Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorised
      IncorrectState
      InvalidLogicalArrayNumber
      TooManyArguments
      InvalidProvenance
      GeneralError
}
```

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

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

IF Cmd.LogicalArrayNumber < 1 OR Cmd.LogicalArrayNumber > MaxLogicalArray THEN
      RETURN InvalidArrayNumber
      EXIT
ENDIF

IF Cmd.NrOfRFPathIDs > 6 THEN
      RETURN TooManyArguments
      EXIT
ENDIF
```

IF Cmd.Provenance NOT IN (Manual, Automatic) THEN
    RETURN InvalidProvenance
    EXIT
ENDIF

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

UNLESS result.allowed THEN
    RETURN result.code
    EXIT
ENDIF

«Store the RF path IDs and provenance for the supplied LogicalArrayNumber and Cmd.PortNumber to non-volatile memory»

IF «the ALD detects a hardware error» THEN
    RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
    RETURN GeneralError
ELSE
    RETURN OK
ENDIF

CommandExit(Cmd.Command, CurrentPort)
EXIT

11.6.7. ADB Get RF Path ID Of Logical Array

Description (Informative):

This command returns the RFPathID list for the requested logical array number, specific to the requesting primary.

Message format (Normative):

```
PrimaryCommand ADBGetRFPathIDOfLogicalArrayCommand {
     CommandCode_t         Command ← 0x0306
     CommandSequence_t     PrimaryCommandSequence
     Subunit_t             Subunit
     DataLength_t          DataLength ← 2
     uint16_t              LogicalArrayNumber
}
```

```
ALDResponse ADBGetRFPathIDOfLogicalArrayResponse {
      CommandCode_t              Command ← 0x0306
      CommandSequence_t          PrimaryCommandSequence
      ReturnCode_t               ReturnCode
      DataLength_t               DataLength
      if (ReturnCode == OK) {
            uint8_t              NrOfRFPathIDs
            uint16_t             RFPathID[NrOfRFPathIDs]
            Provenance_t         Provenance
      }
      else {
            ALDState_t           ALDState
            ConnectionState_t    ConnectionState
      }
}

Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      IncorrectState
      NotAuthorised
      InvalidArrayNumber
}
```

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

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

IF Cmd.LogicalArrayNumber < 1 OR Cmd.LogicalArrayNumber > MaxLogicalArray THEN
      RETURN InvalidLogicalArrayNumber
      EXIT
ENDIF

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

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

RETURN OK, «number of stored RF path IDs and the list of RF path IDs of the requested logical array and its provenance and Cmd.PortNumber»
CommandExit(Cmd.Command, CurrentPort)
EXIT

ANNEX A: Direction definitions for polarisation

This annex shows how the directions from which the antenna is viewed to define its polarisation are defined.

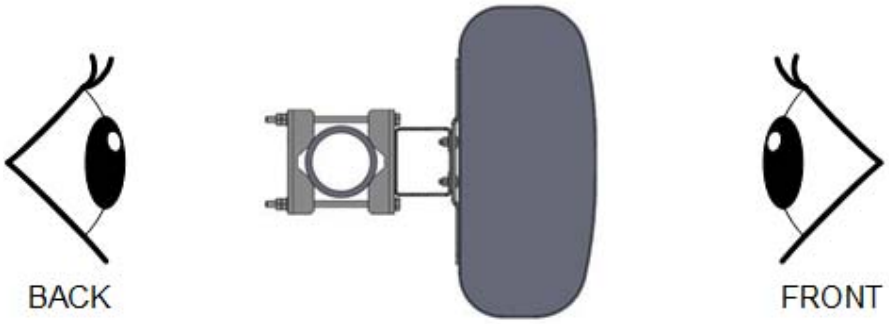

Figure A-1: Array observation reference directions (Back or Front)

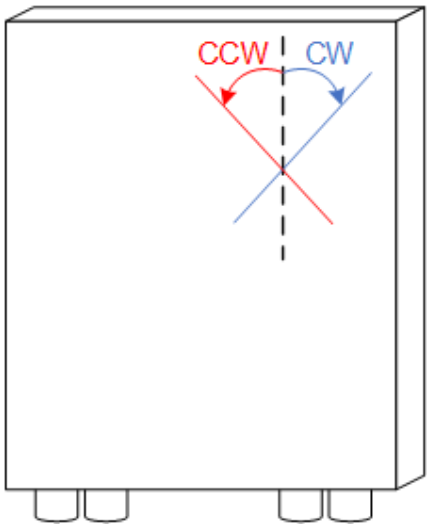

Figure A-2: Array polarisation sense reference (CW (Clockwise) and CCW (Counterclockwise))

ANNEX B: Polarisation comparison example

Vendor X defines slant linear polarisation by looking the antenna from the back and uses clockwise array polarisation sense reference.

Vendor Y defines slant linear polarisation by looking the antenna from the front and uses clockwise array polarisation sense reference.

Vendor X has indicated that Array 1 polarisation is Xpol+. ADB Get Antenna Array Info reports that the vendor uses array observation direction reference “Back” and array polarisation sense reference “CW”.

Vendor Y has indicated that Array 1 polarisation is Xpol−. ADB Get Antenna Array Info reports that the vendor uses array observation direction reference “Front” and array polarisation sense reference “CW”.

Conversion using table 11.5-1:

Vendor X comparable polarisation is B.

Vendor Y comparable polarisation is B.