Remote Electrical Tilt
AISG-ST-RET
vRET3.1.1.7

Revision History

DATE	ISSUE	NOTES
5th November 2018	vRET3.1.1.7	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 document 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 altering the electrical downtilt of an antenna.

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 document and several subunit type standards. This subunit type standard document describes the specific behaviour of the Remote Electrical Tilt (RET) subunit type.

This standard defines the functional behaviour of RET 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: “AISG v3.0”

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

3 AISG-APCC: “Antenna Port Colour Coding Standard”

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:

ALD	Antenna Line Device
DC	Direct Current
MALD	Multi-primary ALD
RET	Remote Electrical Tilt
SALD	Single-primary ALD
TCC	Time-Consuming Command
3GPP	3rd Generation Partnership Project

6. TERMINOLOGY

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

Array	An array is a logical group of single or dual polarized radiators inside the antenna radome supporting a common frequency band and a common beam shape and tilt [3].
Calibrated by design	Intrinsically calibrated, does not allow user initiated calibration.
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.
Mechanical boresight	The axis perpendicular with the antenna aperture.
Jam	A condition in which actuator movement is not possible.

7. DEFINITIONS

uint32_t RETMovementCommandPort

8. GENERAL ASPECTS

8.1. Subunit association

A RET subunit may be associated with several ports without any interconnection between these ports.

8.2. Array

The definition of array means that if the two polarizations of a physical dual polarized array are controlled by one common RET, they are considered to be one dual polarized array.

Furthermore, if the two polarizations of physical dual polarized array are controlled by two independent RETs, they are considered as two independent single polarized arrays.

One RET can control any number of arrays.

8.3. State models

Unless otherwise explicitly allowed, these states shall be retained through a DC power cycle.

8.3.1. Subunit Calibration State model for layer 7

The subunit calibration state model in figure 8.3.1-1 “RETCalState state model” shows the transition between calibration states for a RET subunit.

In certain implementations, a DC power cycle during actuator movement may cause unknown position. In this circumstance the RET shall enter RETNotCalibratedState.

```
Enumeration RETCalState_t : uint8_t {
     RETNotConfiguredState      ← 0
     RETNotCalibratedState      ← 1
     RETCalibratedState         ← 2
}

Persistent Enumeration RETCalState_t RETCalState
```

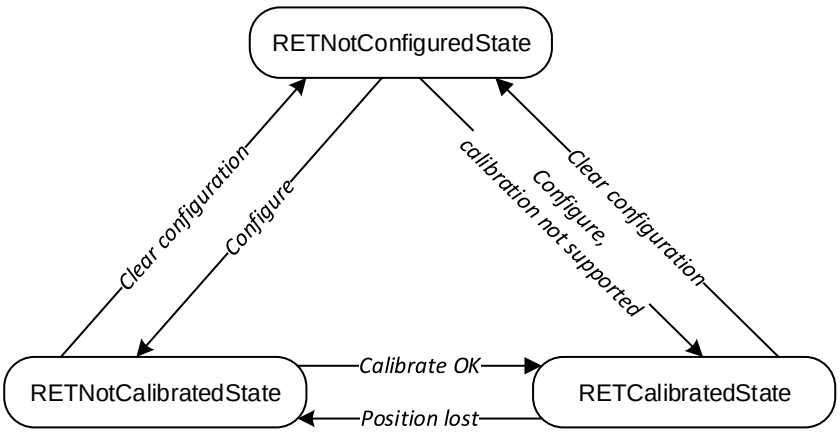

Figure 8.3.1-1 RETCalState state model

8.3.2. Subunit Jam State model for layer 7

The subunit jam state model in figure 8.3.2-1 “RETJamState state model” shows the transition between jam states for a RET subunit.

```
Enumeration RETJamState_t : uint8_t {
     RETNotJammedState    ← 0
     RETJammedState       ← 1
}
persistent Enumeration RETJamState_t RETJamState
```

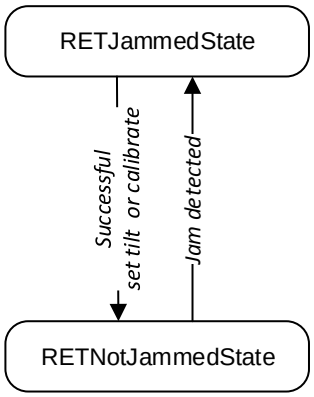

Figure 8.3.2-1 RETJamState state model

8.3.3. RET Moving State model for layer 7

The RET Moving state model in figure 8.3.3-1”RETMovement state model” shows the transition between actuator movement states that are common for all RET subunits in an ALD.

```
Enumeration RETMovementState_t : uint8_t {
     RETNotMovingState    ← 0
     RETMovingState       ← 1
}

Enumeration RETMovementState_t RETMovementState
```

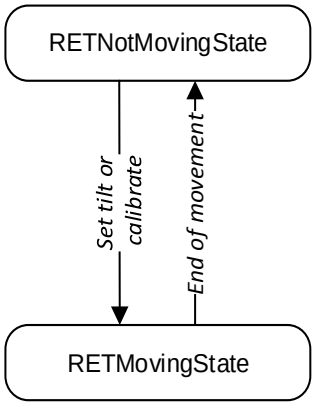

Figure 8.3.3-1 RETMovement state model

8.4. RET capabilities

The RETCapabilities bitfield is set by design.

```
Bitfield RETCapabilities_t : uint8_t {
      GetTiltDuringSetTiltSupported   : Bit 0
      CalibratedByDesign              : Bit 1
      ConfiguredByDesign              : Bit 2
      ReservedForFuture4              : Bit 3    ← 0
      ReservedForFuture3              : Bit 4    ← 0
      ReservedForFuture2              : Bit 5    ← 0
      ReservedForFuture2              : Bit 6    ← 0
      ReservedForFuture2              : Bit 7    ← 0
}

RETCapabilities_t RETCapabilities
```

8.5. Reset

On reset the persistent alarms are raised.

```
ON Reset DO
      FOR I IN 1..NrOfSubunits – 1 DO
            IF Subunits[I].Type = RET THEN
                  IF RETJamState[I] = RETJammedState THEN
                        RAISE AlarmActuatorJammed SEVERITY Major ON SUBUNIT I
                  ENDIF
                  IF RETCalState[I] = RETNotConfiguredState THEN
                        RAISE AlarmNotConfigured SEVERITY Major ON SUBUNIT I
                  ELSEIF RETCalState[I] = RETNotCalibratedState THEN
                        IF CalibratedByDesign[I] THEN
                              RAISE AlarmInternalError SEVERITY Major ON ALD
                        ELSE
                              RAISE AlarmNotCalibrated SEVERITY Major ON SUBUNIT I
                        ENDIF
                  ENDIF
            ENDIF
      DONE
DONE
```

The tilt angle shall be nonvolatile through reset and DC power cycle.

8.6. Actuator movement

This pseudo code defines requirements that must be met all the time during the movement by the implementation specific actuator movement code.

ON position lost DO // Position lost detection is vendor specific
SWITCH RETCalState TO RETNotCalibratedState
SWITCH RETMovementState TO RETNotMovingState
RAISE AlarmNotCalibrated SEVERITY Major ON Cmd.Subunit
SWITCH DCPowerMode TO SteadyStatePowerMode
DONE

ON a jam is detected DO // Jam detection is vendor specific
SWITCH RETJamState TO RETJammedState
SWITCH RETMovementState TO RETNotMovingState
RAISE AlarmActuatorJammed SEVERITY Major ON Cmd.Subunit
SWITCH DCPowerMode TO SteadyStatePowerMode
DONE

ON jam recovered DO
SWITCH RETJamState TO RETNotJammedState
CLEAR AlarmActuatorJammed SEVERITY Major ON Cmd.Subunit
DONE

// Management of DC power mode following jam recovery is implementation specific

ON movement timeout occurred DO
RAISE AlarmMovementTimeout SEVERITY Minor ON Cmd.Subunit
ENDDO

8.7. Return codes

This subunit type standard contains the following subunit type specific reason codes.

```
Enumeration ReturnCode_t : uint16_t {
     RETSetTiltFailed ← 0x0100
}
```

8.8. RET subunit configuration

RET configuration data is used to convert the electrical tilt values to actuator position. This data is specific to particular combinations of actuators and antenna array models.

The RET configuration data can be permanently programmed in the factory, loaded from the antenna when the ALD is installed, or downloaded by the primary. In the first two cases, the RET configuration is unwritable. In the third case, it is writable and its implementation is defined in base standard [1].

A RET with writable RET configuration data is configured by downloading the correct RET configuration file and unconfigured with the RETClearConfig command.

8.9. Resumption of operation

The following data shall be retained after reset:

- Configuration file (if applicable)
- Calibration
- Calibration state
- Tilt value
- Array ID to RET Subunit

9. LAYER 1

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

9.1. DC power consumption

This subunit type standard contains two commands, Set Tilt and Calibrate, which allow the ALD to switch from SteadyStatePowerMode to HighPowerMode.

10. LAYER 2

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

11. LAYER 7

An ALD which contains RET 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**
RET	0x01

Table 11.1-1: Subunit type code

11.2. Overview of commands for RET subunits

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

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

M Mandatory

O Optional

- Not applicable

RET Command	Code	Initiator	Subunit	Timeout	TCC	Mandatory for:			Changes the ConnectionState	Changes to HighPowerMode	Minimum require authority
						Primary	SALD	MALD			
RET Calibrate	0x0100	Primary	>0	4 min	yes	M	M	M	no	yes	RW
RET Set Tilt	0x0101	Primary	>0	2 min	yes	M	M	M	no	yes	RW
RET Get Tilt	0x0102	Primary	>0	1 s	no	M	M	M	no	no	RO
RET Get Capabilities	0x0103	Primary	>0	1 s	no	M	M	M	no	no	RO
RET Set Array ID to Subunit	0x0104	Primary	>0	1 s	no	M	M	M	no	no	RW
RET Get Array ID of Subunit	0x0105	Primary	>0	1 s	no	M	M	M	no	no	RO
RET Clear Config	0x0106	Primary	>0	1 s	no	M	M	M	no	no	RW

Table 11.2-1: Commands for RET subunits

11.3. Tilt representation

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

The tilt accuracy of the actual beam is not defined.

11.4. RET commands

11.4.1. RET Calibrate

Description (Informative):

The calibration command triggers a process during which internal representation of tilt angles is aligned with the physical position of the elevation beam. Some RETs, for example those with absolute position encoders, are intrinsically calibrated and do not require the calibration process to be performed.

Specification (Normative):

The tilt angle on completion of the Calibrate command is vendor specific, so to ensure the correct tilt setting the primary shall perform the Set Tilt command once the calibration is completed.

```
PrimaryCommand RETCalibrateCommand {
      CommandCode_t         Command ← 0x0100
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse RETCalibrateResponse {
      CommandCode_t         Command ← 0x0100
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            RETCalState_t         RETCalState
            RETJamState_t         RETJamState
            RETMovementState_t    RETMovementState
      }
}

Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorised
      IncorrectState
      NotConfigured
      InUseByAnotherPrimary
      CalibrationNotSupported
      CalibrationFailed
}
```

```
IF RETCalState[Cmd.Subunit] = RETNotConfiguredState THEN
    RETURN NotConfigured
    EXIT
ELSEIF RETCapabilities.CalibratedByDesing THEN // For example, uses absolute position
                                                // sensor
    RETURN CalibrationNotSupported
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
LOCK StateLock
IF RETMovementState = RETMovingState THEN
    IF RETMovementCommandPort = CurrentPort THEN
        RETURN Busy
    ELSE
        RETURN InUseByAnotherPrimary
    ENDIF
    UNLOCK StateLock
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF
SWITCH RETMovementState TO RETMovingState
RETMovementCommandPort ← Current Port
UNLOCK StateLock
Switch electronics to consume DC power from the primary which sent the RETCalibrate
command
SWITCH DCPowerMode TO HighPowerMode
Perform calibration
CLEAR AlarmMovementTimeout on Cmd.Subunit
```

IF calibration fails THEN //Raise appropriate alarms
RETURN CalibrationFailed
ELSE
RETURN OK
ENDIF

SWITCH DCPowerMode TO SteadyStatePowerMode
SWITCH RETMovementState TO RETNotMovingState
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.4.2. RET Set Tilt

Description (Informative):

On the receipt of this command the RET subunit sets the tilt of the main beam to the requested angle.

Specification (Normative):

```
PrimaryCommand RETSetTiltCommand {
      CommandCode_t         Command ← 0x0101
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 2
      int16_t               TiltValue
}

ALDResponse RETSetTiltResponse {
      CommandCode_t         Command ← 0x0101
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            RETCalState_t         RETCalState
            RETJamState_t         RETJamState
            RETMovementState_t    RETMovementState
      }
}
```

```
Enumeration ReturnCode_t {
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorised
      IncorrectState
      InUseByAnotherPrimary
      NotCalibrated
      NotConfigured
      OutOfRange

      RETSetTiltFailed
}
```

```
IF the requested tilt angle is not supported THEN
      RETURN OutOfRange
ELSEIF RETCalState = RETNotConfiguredState THEN
      RETURN NotConfigured
ELSEIF RETCalState = RETNotCalibratedState THEN
      RETURN NotCalibrated

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

LOCK StateLock

IF RETMovementState = RETMovingState THEN

      IF RETMovementCommandPort = CurrentPort THEN
            RETURN Busy
      ELSE
            RETURN InUseByAnotherPrimary
      ENDIF

      UNLOCK StateLock
      CommandExit(Cmd.Command, CurrentPort)
      EXIT
ENDIF
```

SWITCH RETMovementState TO RETMovingState
RETMovementCommandPort ← Current Port
UNLOCK StateLock
Switch electronics to consume DC power from the primary which requests the Set Tilt
Switch to HighPowerMode
CLEAR AlarmMovementTimeout on Cmd.Subunit
Perform Tilt change

IF tilt setting fails THEN // Raise appropriate alarms
RETURN RETSetTiltFailed
ELSE
RETURN OK
ENDIF

Switch to SteadyStatePowerMode
SWITCH RETMovementState TO RETNotMovingState
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.4.3. RET Get Tilt

Description (Informative):

On the receipt of this command the RET subunit shall return the current electrical tilt value.

Specification (Normative):

```
PrimaryCommand RETGetTiltCommand {
      CommandCode_t         Command ← 0x0102
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse RETGetTiltResponse {
      CommandCode_t         Command ← 0x0102
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            int16_t               TiltValue
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            RETCalState_t         RETCalState
            RETJamState_t         RETJamState
            RETMovementState_t    RETMovementState
      }
}
```

```
Enumeration ReturnCode_t{
      OK
      FormatError
      UnknownCommand
      Busy
      InvalidSubunitNumber
      InvalidSubunitType
      NotAuthorised
      IncorrectState
      NotCalibrated
      NotConfigured
      GeneralError
}
```

```
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
IF RETCalState = RETNotConfiguredState THEN
      RETURN NotConfigured
ELSEIF RETCalState = RETNotCalibratedState THEN
      RETURN NotCalibrated
ELSEIF current tilt value is unknown THEN
      RETURN GeneralError
ELSE
      RETURN OK, current tilt value
ENDIF
CommandExit(Cmd.Command, CurrentPort)
EXIT
```

11.4.4. RET Get Capabilities

Description (Informative):

On the receipt of this command the RET subunit returns the RET capabilities:

- Whether or not it is possible to query the tilt during a Set Tilt command
- Whether or not it is calibrated by design
- Whether or not it is configured by design
- Supported electrical tilt range

Primaries should use this command whenever the ALD has performed a reset and whenever the RET has been configured (if supported).

Specification (Normative):

```
PrimaryCommand RETGetCapabilitiesCommand {
      CommandCode_t         Command ← 0x0103
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse RETGetCapabilitiesResponse {
      CommandCode_t         Command ← 0x0103
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            int16_t         MinTiltRange
            int16_t         MaxTiltRange
            RETCapabilities_t RETCapabilities
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            RETCalState_t         RETCalState
            RETJamState_t         RETJamState
            RETMovingState_t      RETMovementState
      }
}

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

Primary specification (Normative):

ALD specification (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, tilt range and table of capabilities
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.4.5. RET Set Array ID to Subunit

Description (Informative):

On the receipt of this command the RET subunit stores the provided list of Array IDs.

Specification (Normative):

```
PrimaryCommand RETSetArrayIDToSubunitCommand {
      CommandCode_t         Command ← 0x0104
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength
      uint16_t        NrOfArrayIDs
      for(i = 0; i < NrOfArrayIDs; i++)
      {
            AsciiString ArrayID       // For instance 'R1', 'Y2'
      }
}

ALDResponse RETSetArrayIDToSubunitResponse {
      CommandCode_t         Command ← 0x0104
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t      ALDState
            ConnectionState_t ConnectionState
            RETCalState_t   RETCalState
            RETJamState_t   RETJamState
            RETMovingState_t RETMovementState
      }
}

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

Primary specification (Normative):

ALD specification (Normative):

```
IF ALDType = MALD THEN
      IF ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN
            RETURN NotAuthorised
            EXIT
      ENDIF
ENDIF
IF Cmd.ArrayNumber is not one of 1..MaxArray THEN
      RETURN InvalidArrayNumber
      EXIT
ENDIF
result ← IsCommandAllowed( LIST{      OperatingConnectionState},
                                      Cmd.Command, CurrentPort)
UNLESS result.allowed THEN
      RETURN result.code
      EXIT
ENDIF
Store the Array IDs for the supplied Subunit 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
```

11.4.6. RET Get Array ID of Subunit

Description (Informative):

On the receipt of this command the RET subunit returns the Array ID list.

Specification (Normative):

```
PrimaryCommand RETGetArrayIDOfSubunitCommand {
      CommandCode_t         Command ← 0x0105
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}
```

```
ALDResponse RETGetArrayIDOfSubunitResponse {
      CommandCode_t         Command ← 0x0105
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
            uint16_t NrOfArrayIDs
            for(i = 0; i < NrOfArrayIDs; i++){
               AsciiString ArrayID
            }
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            RETCalState_t         RETCalState
            RETJamState_t         RETJamState
            RETMovingState_t      RETMovementState
      }
}

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

Primary specification (Normative):

ALD specification (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, the supplied RET Subunit, number of stored Array IDs and the list of array IDs of requested RET Subunit
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.4.7. RET Clear Config

Description (Informative):

On the receipt of this command the RET subunit shall clear RET configuration.

Specification (Normative):

```
PrimaryCommand RETClearConfigCommand {
      CommandCode_t         Command ← 0x0106
      CommandSequence_t     PrimaryCommandSequence
      Subunit_t             Subunit
      DataLength_t          DataLength ← 0
}

ALDResponse RETClearConfigResponse {
      CommandCode_t         Command ← 0x0106
      CommandSequence_t     PrimaryCommandSequence
      ReturnCode_t          ReturnCode
      DataLength_t          DataLength
      if (ReturnCode == OK) {
      }
      else {
            ALDState_t            ALDState
            ConnectionState_t     ConnectionState
            RETCalState_t         RETCalState
            RETJamState_t         RETJamState
            RETMovementState_t    RETMovementState
      }
}

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

```
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 RETCapabilities.ConfiguredByDesign THEN
        Erase RET configuration for subunit Cmd.Subunit
        RAISE AlarmNotConfigured SEVERITY Warning ON Cmd.Subunit
        RETURN OK
ELSE
        RETURN ConfigurationNotSupported
ENDIF

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