Remote Electrical Tilt

AISG-ST-RET

vRET3.1.2.1

Revision History

<table>
<thead>
<tr>
<th>DATE</th>
<th>ISSUE</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>27th May 2019</td>
<td>vRET3.1.2.1</td>
<td>Second public release</td>
</tr>
<tr>
<td>5th November 2018</td>
<td>vRET3.1.1.7</td>
<td>First public release</td>
</tr>
</tbody>
</table>

© Copyright AISG Ltd 2017-2019
11.4.6. RET Get Logical Array Numbers Of Subunit....................................................... 27
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 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 standard 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: “Base Standard AISG v3.0”

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

- **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

#include "types.h"

uint16_t RETMovementCommandPort

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

NOTE: The common alarms applicable to a RET are defined in [1], because they are also applicable to other subunit types.

uint16_t NrOfRETSubunits // number of RET subunits within the ALD
8. GENERAL ASPECTS

8.1. Subunit relationship
A RET subunit has a functional relationship with one or more port(s) of the ALD as defined in [1]. In addition, a RET subunit has a functional relationship with one or more logical array(s).

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

One RET subunit can control any number of logical arrays.

8.3. State models

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.

In RETs that support downloading ConfigurationFile and are not CalibratedByDesign, the successful completion of download of file type ConfigurationFile or reception of RecoverFactoryConfiguration command (see [1]) shall cause the RET to enter the RETNotCalibratedState.

This state shall be retained through a DC power cycle.

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

```
PERSISTENT RETCalState_t RETCalState[NrOfRETSubunits]
```

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
}
```

```
Enumeration RETJamState_t RETJamState[NrOfRETSubunits]
```
8.3.3. RET Movement State model for layer 7

The RET Movement 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.

```cpp
Enumeration RETMovementState_t : uint8_t {
    RETNotMovingState ← 0
    RETMovingState ← 1
}
RETMovementState_t RETMovementState[NrOfRETSubunits]
```

8.4. RET capabilities

The RETCapabilities bitfield is set by design.

```cpp
Bitfield RETCapabilities_t : uint8_t {
    GetTiltDuringSetTilt : Bit 0
    CalibratedByDesign   : Bit 1
    ConfiguredByDesign   : Bit 2
}
```
RETCapabilities_t RETCapabilities

8.5. Reset
On reset the persistent alarms are raised.
ON Reset DO
  FOREACH I FROM 1 TO NrOfSubunits – 1 DO
    IF Subunits[I].Type = RET THEN
      IF RETJamState[I] = RETJammedState THEN
        RAISE AlarmActuatorJammed SEVERITY Major ON SUBUNIT I
      ENDIF
      IF RETCapabilities.CalibratedByDesign = 0 THEN
        IF RETCalState[I] = RETNotCalibratedState THEN
          RAISE AlarmNotCalibrated SEVERITY Major ON SUBUNIT I
        ENDIF
      ENDIF
    ENDIF
  ENDF
DONE
DONE

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

8.6. Actuator movement
This pseudocode 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
  RAISE AlarmNotCalibrated SEVERITY Major ON Cmd.Subunit
  IF RETCapabilities.CalibratedByDesign ≠ 0 THEN
    // Replace “Hardware error” with a text describing the problem
    RAISE AlarmGeneralError SEVERITY Major ON Cmd.Subunit, “Hardware error”
  ENDIF
DONE
ON «jam detected» DO // Jam detection is vendor specific
  SWITCH RETJamState TO RETJammedState
  RAISE AlarmActuatorJammed SEVERITY Major ON Cmd.Subunit
DONE
ON «jam recovered» DO
  CLEAR AlarmActuatorJammed SEVERITY Major ON Cmd.Subunit
DONE
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 configuration

Not all RET subunits require configuration. For those which do, vendor specific data is contained within the ALD configuration (see [1]). RET configuration data may include parameters used to convert electrical tilt values to actuator position and may be specific to particular combinations of actuators and antenna models. RET configuration data shall not be used in place of commands defined in these standards.

The configuration data is protected as in table 8.8-1 Configuration data protection.

<table>
<thead>
<tr>
<th>Configuration method</th>
<th>Recovered configuration</th>
<th>Can be overwritten by</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permanently factory configured</td>
<td>Not supported</td>
<td>Nothing</td>
</tr>
<tr>
<td>Factory configured</td>
<td>Factory</td>
<td>File or Auto</td>
</tr>
<tr>
<td>Auto-configured</td>
<td>Auto</td>
<td>File</td>
</tr>
<tr>
<td>Not factory configured</td>
<td>Empty</td>
<td>File</td>
</tr>
</tbody>
</table>

Table 8.8-1: Configuration data protection

NOTE: Auto-configuration is a vendor specific method, in which the antenna contains prestored configuration data, which allows the RET to be automatically configured. Auto-configuration shall take place on after a reset.

8.9. Resumption of operation

The following data shall be retained after reset:

- Configuration file (if applicable)
- Vendor specific calibration details (if applicable)
- Calibration state
- Tilt value
- Assignment of logical array numbers to RET subunit(s)
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 does not define the power consumption of a RET. This subunit type standard contains two commands, RET Set Tilt and RET 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

<table>
<thead>
<tr>
<th>Subunit type</th>
<th>1-octet unsigned integer code</th>
</tr>
</thead>
<tbody>
<tr>
<td>RET</td>
<td>0x01</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>RET commands</th>
<th>Code</th>
<th>Initiator</th>
<th>Subunit</th>
<th>Timeout</th>
<th>TCC</th>
<th>Mandatory for:</th>
<th>Changes to</th>
<th>Changes to</th>
<th>Minimum require authority</th>
</tr>
</thead>
<tbody>
<tr>
<td>RET Calibrate</td>
<td>0x0100</td>
<td>Primary &gt;0</td>
<td>&gt;0</td>
<td>4 min</td>
<td>yes</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>no</td>
</tr>
<tr>
<td>RET Set Tilt</td>
<td>0x0101</td>
<td>Primary &gt;0</td>
<td>&gt;0</td>
<td>2 min</td>
<td>yes</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>no</td>
</tr>
<tr>
<td>RET Get Tilt</td>
<td>0x0102</td>
<td>Primary &gt;0</td>
<td>&gt;0</td>
<td>1 s</td>
<td>no</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>no</td>
</tr>
<tr>
<td>RET Get Capabilities</td>
<td>0x0103</td>
<td>Primary &gt;0</td>
<td>&gt;0</td>
<td>1 s</td>
<td>no</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>no</td>
</tr>
</tbody>
</table>

Site mapping command for RET

| RET Set Logical Array Numbers To Subunit | 0x0104 | Primary >0 | >0 | 1 s | no | O | M | M | no | no | RW |
| RET Get Logical Array Numbers Of Subunit | 0x0105 | Primary >0 | >0 | 1 s | no | O | M | M | no | no | RO |

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 to +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.

Message format (Normative):

The tilt angle on completion of the RET Calibrate command is vendor specific, so to ensure the correct tilt setting the primary shall perform the RET 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
    ALDNotConfigured
    InUseByAnotherPrimary
    CalibrationNotSupported
    CalibrationFailed
}
Primary pseudocode (Normative):
(This section is intentionally left blank)

ALD pseudocode (Normative):
IF ALDState = ALDNotConfiguredState THEN
    RETURN ALDNotConfigured
    EXIT
ELSEIF RETCapabilities.CalibratedByDesign THEN // For example, uses absolute position
    // sensor
    RETURN CalibrationNotSupported
    EXIT
ENDIF
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 ← CurrentPort
    UNLOCK StateLock
    «Switch electronics to consume DC power from the primary which sent the RETCalibrate command»
SWITCH DCPowerMode TO HighPowerMode
CLEAR AlarmMovementTimeout on Cmd.Subunit
«Perform calibration»
IF «calibration fails» THEN  //Raise appropriate alarms
    RETURN CalibrationFailed
ELSE
    RETURN OK
ENDIF
SWITCH RETMovementState TO RETNotMovingState
SWITCH DCPowerMode TO SteadyStatePowerMode
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.

Message format (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
  ALDNotConfigured
  OutOfRange
  RETSetTiltFailed
}

Primary pseudocode (Normative):

(This section is intentionally left blank)

ALD pseudocode (Normative):

IF «the requested tilt angle is not supported» THEN
  RETURN OutOfRange
  EXIT
ELSEIF ALDState = ALDNotConfiguredState THEN
  RETURN ALDNotConfigured
  EXIT
ELSEIF RETCalState = RETNotCalibratedState THEN
  RETURN NotCalibrated
  EXIT
ENDIF

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 ← CurrentPort
UNLOCK StateLock
«Switch electronics to consume DC power from the primary which requests tilting»
SWITCH DCPowerMode 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 RETMovementState TO RETNotMovingState
SWITCH DCPowerMode TO SteadyStatePowerMode
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.

Message format (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
    ALDNotConfigured
}

Primary pseudocode (Normative):
(This section is intentionally left blank)

ALD pseudocode (Normative):

IF ALDType = MALD THEN
    IF ActiveAuth[CurrentPort].Authority[Cmnd.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 ALDState = ALDNotConfiguredState THEN
    RETURN ALDNotConfigured
ELSEIF RETCalState = RETNotCalibratedState THEN
    RETURN NotCalibrated
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 RET 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).

Message format (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 RETMovingState
    }
}

Enumeration ReturnCode_t {
    OK
    FormatError
    UnknownCommand
    Busy
    InvalidSubunitNumber
    InvalidSubunitType
    NotAuthorised
    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,
                                  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 Logical Array Numbers To Subunit

Description (Informative):

This command stores a provided list of logical array numbers the subunit is controlling.

The ALD vendor may have set permanently one or more of the logical array number as read only. If this data is set read only, the command will return return code DataReadOnly. The list of logical array numbers is not stored in that case.

Message format (Normative):

PrimaryCommand RETSetLogicalArrayNumbersToSubunitCommand {
    CommandCode_t Command ← 0x0104
    CommandSequence_t PrimaryCommandSequence
    Subunit_t Subunit
    DataLength_t DataLength
    uint8_t NrOfLogicalArrays
    uint16_t LogicalArrayNumbers[NrOfLogicalArrays]
}
ALDResponse RETSetLogicalArrayNumbersToSubunitResponse {
    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
    DataReadOnly
    InvalidLogicalArrayNumber
    GeneralError
}

**Primary pseudocode (Normative):**

*(This section is intentionally left blank)*

**ALD pseudocode (Normative):**

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

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

UNLESS result.allowed THEN
    RETURN result.code
    EXIT
ENDIF
IF «Logical array number not read only in this RET subunit» THEN
   «Store the logical array numbers for the supplied Subunit to non-volatile memory»
ELSE
   RETURN DataReadOnly
   EXIT
ENDIF

IF «the ALD detects a hardware error» THEN
   // Replace “Hardware error” with a text describing the problem
   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 Logical Array Numbers Of Subunit

Description (Informative):
This command returns the list of logical arrays the subunit is controlling.

Message format (Normative):

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

ALDResponse RETGetLogicalArrayNumbersOfSubunitResponse {
   CommandCode_t Command ← 0x0105
   CommandSequence_t PrimaryCommandSequence
   ReturnCode_t ReturnCode
   DataLength_t DataLength
   if (ReturnCode == OK) {
      uint8_t NrOfLogicalArrays
      uint16_t LogicalArrayNumber[NrOfLogicalArrays]
   }
   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
}

Primary pseudocode (Nvormative):
(This section is intentionally left blank)

ALD pseudocode (Normative):

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

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

UNLESS result.allowed THEN
  RETURN result.code
ENDIF

RETURN OK, « number of stored logical arrays and the list of array numbers of requested RET Subunit»
CommandExit(Cmd.Command, CurrentPort)
EXIT