Tower Mounted Low Noise Amplifier
AISG-ST-TMA
vTMA3.0.0.6

Revision History

<table>
<thead>
<tr>
<th>DATE</th>
<th>ISSUE</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>5th November 2018</td>
<td>vTMA3.0.0.6</td>
<td>First public release.</td>
</tr>
</tbody>
</table>

© Copyright AISG Ltd 2018
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 tower mounted amplifiers (sometimes referred to as masthead or ground mounted amplifiers).

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 standards.
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 tower mounted amplifier (TMA) subunit type.

This standard defines the functional behaviour of the TMA 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”
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
- LNA   Low Noise Amplifier
- MALD  Multi-primary ALD
- RF    Radio Frequency
- SALD  Single-primary ALD
- TCC   Time-Consuming Command
- TMA   Tower Mounted Amplifier
- 3GPP  3rd Generation Partnership Project
6. TERMINOLOGY

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

- **Gain**: A factor by which power is increased in an amplifier, usually expressed in dB.
- **LNA path**: A path including LNA and possible bypass switches.
- **Low noise amplifier**: An amplifier that amplifies a very low power signal without significantly degrading its signal-to-noise ratio.
- **TMA bypass**: An RF path around an LNA to provide alternative route.
- **TMA bypass path**: An RF path around an LNA including bypass switches.
- **TMA bypass switch**: A functionality that switches between normal and bypass mode.
- **Tower mounted amplifier**: A unit typically providing uplink and downlink path and containing an LNA and associated filters.
7. DEFINITIONS

GainTable shall be initialised on reset to contain the possible gain settings of the TMA.

```c
uint8_t NrOfGainRanges
GainRange_t GainTable[NrOfGainRanges]
```

`TMAMode_t` is used to identify the normal or bypass mode of TMA.

```c
Enumeration TMAMode_t: uint8_t {
    BypassMode ← 0
    NormalMode ← 1
}
```

```c
Enumeration TMAAlarmCode_t : uint16_t {
    TMAAlarmLNAFailed ← 0x0200
    TMAAlarmBypassFailed ← 0x0201
    TMAAlarmNoRXConnection ← 0x0202
}
```
8. GENERAL ASPECTS

8.1. Subunit association

A TMA subunit may be associated with two or more RF ports with interconnections between some or all of these ports.

8.2. State models

8.2.1. State model for layer 7

The state model in figure 8.2.1-1 “TMAState state model” shows the relationship between different states of the TMA.

In TMANormalState the TMA is fully operational and amplifies the signal.

In TMACommandedBypassState the TMA is in BypassMode because it was commanded by the primary to do so.

In TMALNAPathImpairedNormalState the TMA is operational but with reduced amplification performance.

In TMALNAPathImpairedBypassState the TMA is operational but with reduced amplification performance, commanded by primary to BypassMode.

In TMALNAPathFailedNormalState the TMA is unable to amplify the signal and is not in BypassMode.

In TMALNAPathFailedBypassState the TMA is unable to amplify the signal and has autonomously switched to BypassMode.

In NoRXConnectionState the TMA is unable to amplify the signal via LNA path and is unable to transfer the signal through TMA bypass path.

Enumeration TMAState_t : uint8_t {
  TMANormalState ← 0
  TMACommandedBypassState ← 1
  TMALNAPathImpairedNormalState ← 2
  TMALNAPathImpairedBypassState ← 3
  TMALNAPathFailedNormalState ← 4
  TMALNAPathFailedBypassState ← 5
  NoRXConnectionState ← 6
}

TMAState_t TMAState
8.3. TMA capabilities

The TMACapabilities bitfield shall be initialised during reset.

```c
Bitfield TMACapabilities_t : uint8_t {
    LNADiagnostic : Bit 0  // Able to diagnose LNA failure
    Bypass        : Bit 1
    RedundantLNA  : Bit 2  // Multiple redundant amplifiers
    BypassDiagnostic : Bit 3  // Able to diagnose bypass switch failure
    AdjustableGain : Bit 4
    ReservedForFuture3 : Bit 5 ← 0
    ReservedForFuture2 : Bit 6 ← 0
    ReservedForFuture1 : Bit 7 ← 0
}
TMACapabilities_t TMACapabilities
```

8.4. Reset

On reset persistent alarms are not raised because they are handled by LNA monitoring, and they will be raised almost immediately.

8.5. LNA monitoring

This pseudo code defines requirements that must be met all the time during the operation of the TMA if the TMA capabilities are supported.
The details of hardware interaction are vendor specific.

FOR I IN 1..NrOfSubunits – 1 DO
    IF Subunits[I].Type = TMA THEN
        IF TMAState = TMALNANormalState
            AND (TMACapabilities.LNADiagnostic AND amplifier completely fails
            OR TMACapabilities.BypassDiagnostic AND LNA path broken) THEN
                TMAState ← TMALNAPathFailedNormalState
                RAISE TMAAlarmLNAFailed SEVERITY Major on Cmd.Subunit,
                “LNA path completely failed”
            ENDIF
        FI
        IF TMAState = TMALNANormalState
            AND TMACapabilities.LNARedundantLNA
            AND TMACapabilities.LNADiagnostic
            AND amplifier partially fails THEN
            TMAState ← TMALNAPathImpairedNormalState
            RAISE TMAAlarmLNAFailed SEVERITY Minor on Cmd.Subunit,
            “LNA path impaired”
        ENDIF
        IF TMAState = TMALNAPathImpairedNormalState
            AND TMACapabilities.LNADiagnostic
            AND amplifier completely fails THEN
            TMAState ← TMALNAPathFailedNormalState
            RAISE TMAAlarmLNAFailed SEVERITY Major on Cmd.Subunit,
            “LNA path completely failed”
        ENDIF
        IF TMAState = TMALNAPathImpairedImpairedNormalState
            AND TMACapabilities.Bypass THEN
            Move TMA to BypassMode
            TMAState ← TMALNAPathFailedBypassState
            RAISE TMAAlarmLNAFailed SEVERITY Major on Cmd.Subunit,
            “LNA path failed”
        ENDIF
    FI
ENDFOR
IF TMAState = TMALNAPathFailedNormalState
    AND TMACapabilities.LNADiagnostic
    AND amplifier is working THEN
        TMAState ← TMANormalState
        CLEAR TMAAlarmLNAFailed ON Cmd.Subunit
    ENDIF

IF TMAState = TMALNAPathFailedNormalState
    AND TMACapabilities.RedundantLNA
    AND TMACapabilities.RedundantLNA
    AND amplifier partially fails THEN
        TMAState ← TMALNAPathImpairedNormalState
        RAISE TMAAlarmLNAFailed SEVERITY Minor on Cmd.Subunit,
        "LNA path impaired"
    ENDIF

IF TMAState = TMACommandedBypassState
    AND TMACapabilities.LNADiagnostic
    AND amplifier completely fails
    AND TMACapabilities.BypassDiagnostic
    AND Bypass path is broken THEN
        TMAState ← NoRXConnectionState
        RAISE TMAAlarmNoRXConnection SEVERITY Major on Cmd.Subunit,
        "LNA path and bypass path failed"
    ENDIF

IF TMAState = TMALNAImpairedBypassState
    AND TMACapabilities.LNADiagnostic
    AND amplifier completely fails
    AND TMACapabilities.BypassDiagnostic
    AND Bypass path is broken THEN
        TMAState ← NoRXConnectionState
        CLEAR TMAAlarmLNAFailed ON Cmd.Subunit
        RAISE TMAAlarmNoRXConnection SEVERITY Major on Cmd.Subunit,
        "LNA path and bypass path failed"
    ENDIF

IF TMAState = TMALNAImpairedBypassState
    AND TMACapabilities.LNADiagnostic
    AND amplifier partially fails or is working
    AND TMACapabilities.BypassDiagnostic
    AND Bypass path is broken THEN
        TMAState ← TMALNAPathImpairedNormalState
        CLEAR TMAAlarmLNAFailed ON Cmd.Subunit
        RAISE TMAAlarmBypassFailed SEVERITY Minor on Cmd.Subunit,
        "LNA path impaired or is working and bypass path is broken"
    ENDIF
IF TMAState = TMACommandedBypassState
    AND TMACapabilities.BypassDiagnostic
    AND Bypass path is broken THEN
        TMAState ← TMANormalState
        RAISE TMAAlarmBypassFailure SEVERITY Minor on Cmd.Subunit,  
        "Bypass failed"
ENDIF

IF TMAState = NoRXConnectionState
    AND TMACapabilities.BypassDiagnostic
    AND Bypass path is working THEN
        TMAState ← TMALNAPathFailedBypassState
        CLEAR TMAAlarmNoRXConnection ON Cmd.Subunit
        RAISE TMAAlarmLNAFailure SEVERITY Major on Cmd.Subunit,  
        “LNA failed”
ENDIF

IF TMAState = TMALNAPathFailedBypassState
    AND TMACapabilities.LNADiagnostic
    AND amplifier partially fails or is working THEN
        TMAState ← TMALNAPathFailedNormalState
        RAISE TMAAlarmLNAFailure SEVERITY Major on Cmd.Subunit,  
        “Bypass failed”
ENDIF

DONE

8.6. Return codes
This subunit type standard contains the following subunit type specific return codes.

Enumeration ReturnCode_t : uint16_t {
    TMAUnsupportedGainValue ← 0x0200
    TMAMajorFault ← 0x0201
    TMAMinorFault ← 0x0202
    TMANotPredictableGain ← 0x0203
}

8.7. Resumption of operation
The following data shall be retained after reset:

- TMA Mode
- Gain setting
9. LAYER 1

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

9.1. DC power consumption

This subunit type standard does not define the power consumption of a TMA. 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 SleepPowerMode.
10. LAYER 2
All definitions and standards for ALDs in [1] regarding layer 2 shall be valid for ALDs which contain TMA subunits.
11. LAYER 7

An ALD which contains TMA 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>TMA</td>
<td>0x02</td>
</tr>
</tbody>
</table>

Table 11.1-1: Subunit type code

11.2. Overview of commands for TMA subunits

The table below shows an overview of all commands used in this TMA subunit type standard. The following abbreviations are used in the Table 11.2-1 "Commands for TMA subunits":

M  Mandatory
O  Optional
-  Not applicable

<table>
<thead>
<tr>
<th>TMA Command</th>
<th>Code</th>
<th>Initiator</th>
<th>Subunit</th>
<th>Timeout</th>
<th>TCC</th>
<th>Mandatory for:</th>
<th>Changes the ConnectionState</th>
<th>Changes the PowerMode</th>
<th>Minimum required authority</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMA Get Capabilities</td>
<td>0x0200</td>
<td>Primary</td>
<td>&gt;0</td>
<td>1 s</td>
<td>no</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>RO</td>
</tr>
<tr>
<td>TMA Set Mode</td>
<td>0x0201</td>
<td>Primary</td>
<td>&gt;0</td>
<td>1 s</td>
<td>no</td>
<td>M</td>
<td>O</td>
<td>O</td>
<td>RW</td>
</tr>
<tr>
<td>TMA Get Mode</td>
<td>0x0202</td>
<td>Primary</td>
<td>&gt;0</td>
<td>1 s</td>
<td>no</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>RO</td>
</tr>
<tr>
<td>TMA Set Gain</td>
<td>0x0203</td>
<td>Primary</td>
<td>&gt;0</td>
<td>1 s</td>
<td>no</td>
<td>M</td>
<td>O</td>
<td>O</td>
<td>RW</td>
</tr>
<tr>
<td>TMA Get Gain</td>
<td>0x0204</td>
<td>Primary</td>
<td>&gt;0</td>
<td>1 s</td>
<td>no</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>RO</td>
</tr>
</tbody>
</table>

Table 11.2-1: Commands for TMA subunits

11.3. TMA commands

11.3.1. TMA Get Capabilities

Description (Informative):

On receipt of the command, the ALD shall respond with the capabilities and parameters indicating the functionalities of the addressed TMA subunit.
The supported gain ranges shall follow the definitions in [1]. Gain value in bypass state shall not be reported as part of the gain ranges.

NOTE: Bit value 0 represents function is not supported. Bit value 1 represents function is supported.

**Specification (Normative):**

```c
PrimaryCommand TMAGetCapabilitiesCommand {
    CommandCode_t Command ← 0x0200
    CommandSequence_t PrimaryCommandSequence
    Subunit_t Subunit
    DataLength_t DataLength ← 0
}

ALDResponse TMAGetCapabilitiesResponse {
    CommandCode_t Command ← 0x0200
    CommandSequence_t PrimaryCommandSequence
    ReturnCode_t ReturnCode
    DataLength_t DataLength
    if (ReturnCode == OK) {
        TMACapabilities_t TMACapabilities
        GainRange_t Ranges[NrOfGainRanges]
        uint8_t NrOfFrequencyRanges
        FrequencyRange_t Ranges[NrOfFrequencyRanges]
    } else {
        ALDState_t ALDState
        ConnectionState_t ConnectionState
    }
}
```

**Enumeration ReturnCode_t:**

- OK
- FormatError
- Busy
- InvalidSubunitNumber
- InvalidSubunitType
- IncorrectState
- UnknownCommand
- NotAuthorised

**IF ALDType = MALD**

**AND ActiveAuth[CurrentPort].Authority[Cmd.Subunit] = NoAccess THEN**

**RETURN NotAuthorised**

**EXIT**

**ENDIF**

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

**UNLESS result.allowed THEN**

**RETURN result.code**

**EXIT**

**ENDIF**
RETURN OK, TMA Capabilities, NrOfGainRanges, LIST for every gain range(Min, Max, StepSize), NrOfFrequencyRanges, LIST for every frequency range(Min, Max)
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.3.2. TMA Set Mode

Description (Informative):
The TMA Set Mode procedure shall only be supported if the TMA subunit can be set in bypass mode. On receipt of the command, the ALD shall first set the TMA subunit in the requested mode, and then return a response message. If a TMA subunit in bypass mode receives the command TMASetMode to BypassMode, the command shall not be performed but the response OK shall be returned. Similarly, if a TMA subunit in normal mode receives the command TMASetMode to NormalMode, the command shall not be performed but the response OK shall be returned.

Specification (Normative):

PrimaryCommand TMASetModeCommand {
    CommandCode_t Command ← 0x0201
    CommandSequence_t PrimaryCommandSequence
    Subunit_t Subunit
    DataLength_t DataLength ← 1
    TMAMode_t TMAMode
}

ALDResponse TMASetModeResponse {
    CommandCode_t Command ← 0x0201
    CommandSequence_t PrimaryCommandSequence
    ReturnCode_t ReturnCode
    DataLength_t DataLength
    if (ReturnCode == OK) {
    }
    Else {
        ALDState_t ALDState
        ConnectionState_t ConnectionState
    }
}

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

UNLESS TMA Capabilities.Bypass THEN
    RETURN UnsupportedCommand
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

IF Cmd.TMAMode = NormalMode THEN
    IF TMASState = TMACommandedBypassState
        OR TMASState = TMANormalImpairedBypassState THEN
        Initiate action to change bypass switch to normal mode
    ENDIF
ELSEIF Cmd.TMAMode = BypassMode THEN
    IF TMASState = TMANormalState
        OR TMASState = TMANormalImpairedNormalState THEN
        Initiate action to change bypass switch to bypass mode
    ENDIF
ELSE
    RETURN UnsupportedValue
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

RETURN OK
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.3.3. TMA Get Mode

Description (Informative):
On receipt of the command, the ALD shall respond with the mode flag indicating whether the TMA subunit is in NormalMode or in BypassMode. TMA subunits which do not support BypassMode shall return NormalMode.

Specification (Normative):
PrimaryCommand TMAGetModeCommand {
    CommandCode_t Command ← 0x02
    CommandSequence_t PrimaryCommandSequence
    Subunit_t Subunit
    DataLength_t DataLength ← 0
}
ALDResponse TMAGetModeResponse {
    CommandType_t Command ← 0x0202
    CommandSequence_t PrimaryCommandSequence
    ReturnCode_t ReturnCode
    DataLength_t DataLength
    if (ReturnCode == OK) {
        TMAMode_t TMAMode
    } else {
        ALDState_t ALDState
        ConnectionState_t ConnectionState
    }
}

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

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, TMAMode
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.3.4. TMA Set Gain

Description (Informative):

On receipt of the command (if supported), the ALD shall first set the addressed TMA subunit
to the gain determined by the TMA gain figure parameter, and then return the response
message. The TMA gain figure parameter is calculated as 10 times the required gain
expressed in dB. This encoding allows the gain to be set with a resolution of 0.1 dB while
using an integer parameter.

If the TMA subunit is set in bypass mode by TMASetMode, and TMASetGain is received,
then the procedure shall be performed and bypass mode shall be retained.
Gain shall be accepted if the requested gain value is supported.
For unsupported values the TMA subunit shall respond TMAUnsupportedGainValue.

**Specification (Normative):**

```c
PrimaryCommand TMASetGainCommand {
    CommandCode_t Command ← 0x0203
    CommandSequence_t PrimaryCommandSequence
    Subunit_t Subunit
    DataLength_t DataLength ← 2
db_t Gain
}

ALDResponse TMASetGainResponse {
    CommandCode_t Command ← 0x0203
    CommandSequence_t PrimaryCommandSequence
    ReturnCode_t ReturnCode
    DataLength_t DataLength
    if (ReturnCode == OK) {
    } else {
        ALDState_t ALDState
        ConnectionState_t ConnectionState
    }
}

Enumeration ReturnCode_t {
    OK
    FormatError
    Busy
    IncorrectState
    InvalidSubunitNumber
    InvalidSubunitType
    TMAUnsupportedGainValue
    UnknownCommand
    NotAuthorised
    TMAMajorFault
    TMAMinorFault
}

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 TMA is in Bypass mode due to major TMA fault THEN
    RETURN TMAMajorFault
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

IF TMA is in NormalMode AND gain setting cannot be achieved THEN // This is defined by the vendor
    RETURN TMAMinorFault
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

FOREACH I FROM 0 TO NrOfGainRanges -1 DO
    NEXT IF Cmd.Gain > GainTable[I].Max
    IF Cmd.Gain < GainTable[I].Min THEN
        RETURN TMAUnsupportedGainValue
        // Special case covering
        // GainTable[I].Stepsize = 0
    ELSEIF Cmd.Gain = GainTable[I].Min
        // Special case covering
        // GainTable[I].Stepsize = 0
        OR (Cmd.Gain - GainTable[I].Min) MOD GainTable[I].StepSize = 0 THEN
        Set TMA gain to Cmd.Gain
        RETURN OK
    ELSE
        RETURN TMAUnsupportedGainValue
    ENDIF
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
DONE

RETURN TMAUnsupportedGainValue
CommandExit(Cmd.Command, CurrentPort)
EXIT

11.3.5. TMA Get Gain

Description (Informative):
On receipt of the command, the ALD shall return the set gain of the TMA subunit. EstimatedGain is used to represent the actual gain. In NormalMode without a TMA fault, the EstimatedGain shall be equal to the SetGain. In BypassMode and in case of a TMA fault in NormalMode, the EstimatedGain shall be a prediction of the actual gain.

Specification (Normative):

PrimaryCommand TMAGetGainCommand {
        CommandCode_t Command ← 0x0204
        CommandSequence_t PrimaryCommandSequence
        Subunit_t Subunit
        DataLength_t DataLength ← 0
    }
ALDResponse TMAGetGainResponse {
    CommandCode_t Command ← 0x0204
    CommandSequence_t PrimaryCommandSequence
    ReturnCode_t ReturnCode
    DataLength_t DataLength
    if (ReturnCode == OK) {
        dB_t SetGain
        dB_t EstimatedGain
        TMAMode_t TMAMode
    } else {
        ALDState_t ALDState
        ConnectionState_t ConnectionState
    }
}

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

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

IF gain can be determined THEN // This is defined by the
    RETURN OK, EstimatedGain, TMAMode
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ELSE
    RETURN TMANotPredictableGain
    CommandExit(Cmd.Command, CurrentPort)
    EXIT
ENDIF

RETURN OK, SetGain
CommandExit(Cmd.Command, CurrentPort)
EXIT