



**Society of Cable  
Telecommunications  
Engineers**

---

**ENGINEERING COMMITTEE  
HFC Management Subcommittee**

---

**AMERICAN NATIONAL STANDARD**

**ANSI/SCTE 38-1 2009**

**Hybrid Fiber/Coax Outside Plant Status Monitoring  
SCTE-HMS-PROPERTY-MIB  
Management Information Base (MIB) Definition**

## NOTICE

The Society of Cable Telecommunications Engineers (SCTE) Standards are intended to serve the public interest by providing specifications, test methods and procedures that promote uniformity of product, interchangeability and ultimately the long term reliability of broadband communications facilities. These documents shall not in any way preclude any member or nonmember of SCTE from manufacturing or selling products not conforming to such documents, nor shall the existence of such standards preclude their voluntary use by those other than SCTE members, whether used domestically or internationally.

SCTE assumes no obligations or liability whatsoever to any party who may adopt the Standards. Such adopting party assumes all risks associated with adoption of these Standards or Recommended Practices, and accepts full responsibility for any damage and/or claims arising from the adoption of such Standards or Recommended Practices.

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. SCTE shall not be responsible for identifying patents for which a license may be required or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Patent holders who believe that they hold patents which are essential to the implementation of this standard have been requested to provide information about those patents and any related licensing terms and conditions. Any such declarations made before or after publication of this document are available on the SCTE web site at <http://www.scte.org>.

All Rights Reserved  
© Society of Cable Telecommunications Engineers, Inc. 2009  
140 Philips Road  
Exton, PA 19341

# CONTENTS

<b>SCOPE</b> .....	<b>1</b>
<b>COPYRIGHT</b> .....	<b>1</b>
<b>NORMATIVE REFERENCE</b> .....	<b>1</b>
<b>INFORMATIVE REFERENCE</b> .....	<b>1</b>
<b>TERMS AND DEFINITIONS</b> .....	<b>1</b>
<b>REQUIREMENTS</b> .....	<b>1</b>

## **SCOPE**

This document defines the "properties" that may be associated with each parameter in HMS MIBs.

## **COPYRIGHT**

The MIB definition found in this document may be incorporated directly in products without further permission from the copyright owner, SCTE.

## **NORMATIVE REFERENCE**

IETF RFC 1155

SCTE 37

## **INFORMATIVE REFERENCE**

None

## **TERMS AND DEFINITIONS**

This document defines the following terms:

**Management Information Base (MIB)** – the specification of information in a manner that allows standard access through a network management protocol.

## **REQUIREMENTS**

This section defines the mandatory syntax of the SCTE-HMS-PROPERTY-MIB. It follows the IETF Simple Network Management Protocol (SNMP) for defining the managed objects.

The syntax is given below.

-- Module Name: HMS026R16.MIB (SCTE 38-1)

SCTE-HMS-PROPERTY-MIB DEFINITIONS ::= BEGIN

IMPORTS

Integer32, MODULE-IDENTITY, OBJECT-TYPE  
FROM SNMPv2-SMI  
OBJECT-GROUP, MODULE-COMPLIANCE  
FROM SNMPv2-CONF  
propertyIdent  
FROM SCTE-HMS-ROOTS; -- see HMS072

propertyModuleIdentity MODULE-IDENTITY

LAST-UPDATED "200403290000Z" -- March 29, 2004

ORGANIZATION "SCTE HMS Working Group"

CONTACT-INFO

"Hung Nguyen,  
SCTE HMS Subcommittee, Chairman  
Time Warner Cable  
mailto:standards@scte.org"

DESCRIPTION

"This MIB contains information that must be supported by all HMS network elements, including but not limited to, transponders, line monitors, amplifiers, fiber nodes, and power supplies.

The Property MIB defines the 'properties' that may be associated with each parameter. This MIB is defined so that these 'properties' may be applied to any parameter, because the index to the MIB is the object identifier of the parameter. The purpose of a 'property' is to provide a mechanism to manage alarm thresholds. It is not the responsibility of the transponder to check for violation of the above recommendations. The element manager is responsible for checking alarm limit values.

Entries in the property table are specifically for 'analog' parameters. The discrete property table is used to monitor other parameters.

Each property entry has four alarm threshold levels that may be established.  
These are:

LOLO	Alarm threshold for the extreme low condition.
LO	Alarm threshold for the low condition.
HI	Alarm threshold for the high condition.
HIHI	Alarm threshold for the extreme high condition.

In addition, there is a 'Deadband' setting which applies to all alarm thresholds. After an alarm occurs, the parameter value must pass back over the alarm threshold by this amount for the alarm condition to be cleared. This Deadband is smaller than the distance between any two alarm thresholds to avoid indeterminate states.

Alarm detection for each threshold is controlled by a specific bit in the alarmEnable variable for the entry. Alarm detection is active when the corresponding bit in alarmEnable is enabled.

When an alarm condition is detected, in either the propertyTable or the discretePropertyTable, an entry is created in the alarm log ( see HMS023Rx.MIB ) and an alarmEvent SNMP trap sent by the transponder/agent.

#### NOTE

Parameters which do not 'exist' must NOT have properties that are accessible. For example, in the HMS027 MIB (SCTE 38-4), the MIB object psOutputPowerSupport indicates whether or not the power supply supports the psPowerOut object. If the psPowerOut object is NOT supported, then the properties normally associated with the psPowerOut object must not be accessible.

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller) values. Values outside of the supported range will return a bad value error.

"

REVISION "200403290000Z" -- March 29, 2004

DESCRIPTION

"

1. SCTE-HMS-PROPERTY-MIB MIB module converted to SMIV2 syntax so as to define MODULE-COMPLIANCE groups. These groups need to be referred to by other MIB modules such as those defined by HMS Indoor Optics.

The publishing of this MIB in SMIV2 does not imply the agent implementation must use a particular version of SNMP (the SNMP agent could communicate using SNMPv1, v2c or v3).

2. Clarification made in the DESCRIPTION clause on usage of the MIB variable currentAlarmOID.

This version published as 'HMS026R16.mib' (SCTE 38-1)  
This version obsoletes 'HMS026R14.mib' .

..

::= { propertyIdent 4 }

propertyMIBConformance OBJECT IDENTIFIER ::= { propertyModuleIdentity 1 }  
propertyMIBCompliances OBJECT IDENTIFIER ::= { propertyMIBConformance 1 }  
propertyMIBGroups OBJECT IDENTIFIER ::= { propertyMIBConformance 2 }

propertyTable OBJECT-TYPE  
SYNTAX SEQUENCE OF PropertyEntry  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION  
"A table that contains information about NE parameter properties."  
::= { propertyIdent 1 }

propertyEntry OBJECT-TYPE  
SYNTAX PropertyEntry  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION  
"A list of information about each property.

The OID suffix for an entry in this table is constructed by appending the length of parameterOID and then the components of parameterOID to identify an instance.

The first two components of parameterOID will generally be

1.3 which will be encoded separately as 1 and 3, not as a single value of 43 (decimal). When parameterOID identifies a scalar object, it is expected that the final suffix of .0 will be included.

If parameterOID was 1.3.5.0, then 4.1.3.5.0 are the resulting components of the OID suffix."

```
INDEX { parameterOID }  
::= { propertyTable 1 }
```

```
PropertyEntry ::= SEQUENCE {  
    parameterOID  
        OBJECT IDENTIFIER,  
    alarmEnable  
        OCTET STRING,  
    currentAlarmState  
        INTEGER,  
    analogAlarmHHHI  
        Integer32,  
    analogAlarmHI  
        Integer32,  
    analogAlarmLO  
        Integer32,  
    analogAlarmLOLO  
        Integer32,  
    analogAlarmDeadband  
        Integer32  
}
```

```
parameterOID OBJECT-TYPE  
SYNTAX OBJECT IDENTIFIER  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION
```

"Index into propertyTable. This is the OID of the parameter whose property is being accessed.

Example: OID of power supply the first instance of the  
psInputVoltage is 1.3.6.1.4.1.5591.1.4.2.1.23.1"  
::= { propertyEntry 1 }

alarmEnable OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(1))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Alarm enable bit mask. A 1 in a bit position indicates the alarm  
is enabled.

Bit 0 = LOLO (Major alarm)

Bit 1 = LO (Minor alarm)

Bit 2 = HI (Minor alarm)

Bit 3 = HIHI (Major alarm)

Bit 4 = Unused, must be zero

Bit 5 = Unused, must be zero

Bit 6 = Unused, must be zero

Bit 7 = Unused, must be zero

This object should be kept in NV memory"

::= { propertyEntry 2 }

currentAlarmState OBJECT-TYPE

SYNTAX INTEGER {

casNominal (1),

casHIHI (2),

casHI (3),

casLO (4),

casLOLO (5)

}

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The object contains the current alarm status associated with this property entry."

::= { propertyEntry 3 }

analogAlarmHIHI OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The HIHI (Major alarm) alarm occurs at this value. The unit associated with this property is the same as that of the parameter addressed.  
This object should be kept in NV memory.

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller) values. Values outside of the supported range will return a bad value error."

::= { propertyEntry 4 }

analogAlarmHI OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The HI (Minor alarm) alarm occurs at this value. The unit associated with this property is the same as that of the parameter addressed.  
This object should be kept in NV memory.

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller) values. Values outside of the supported range will return a bad value error."

::= { propertyEntry 5 }

analogAlarmLO OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The LO (Minor alarm) alarm occurs at this value. The unit associated with this property is the same as that of the parameter addressed.  
This object should be kept in NV memory.

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller) values. Values outside of the supported range will return a bad value error."

::= { propertyEntry 6 }

#### analogAlarmLOLO OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The LOLO (Major alarm)alarm occurs at this value. The unit associated with this property is the same as that of the parameter addressed.

This object should be kept in NV memory.

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller) values. Values outside of the supported range will return a bad value error."

::= { propertyEntry 7 }

#### analogAlarmDeadband OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Deadband for prevention of alarm oscillation. An alarm does not return to normal until the value either

(a) passes the original threshold by this amount in the opposite direction of the alarm, or

(b) alarm is disabled.

This item should be an unsigned integer.

This property is in the same engineering units as the parameter for which it belongs.

This object should be kept in NV memory.

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller) values. Values outside of the supported range will return a bad value error."

::= { propertyEntry 9 }

-- The following table contains zero or more entries which are the  
 -- alarms that are currently "active" for a NE.  
 -- The motivation behind the table is to minimize the number of  
 -- transactions between a management application and a NE to retrieve  
 -- it's current alarm state. Current alarm information is available in  
 -- the propertyTable, however there is a probability in the nominal case  
 -- that few alarms will be active at any given moment in time. Given that  
 -- the propertyTable is of fixed length, iteration through an NE's entire  
 -- table is required to confirm this. Since this table only contains  
 -- entries corresponding to those entries in the property which have  
 -- active alarms, the table is usually empty.  
 -- It is suggested that a management application use an iterative  
 -- algorithm, employing GET-NEXT to retrieve information from this table.  
 -- The algorithm starts with the object identifier, currentAlarmTable, without  
 -- an index specification. The iteration continues until an object value is  
 -- returned which is not a member of the table.  
 -- This should usually occur immediately. If any alarms  
 -- are active, then values are returned for only those properties which have  
 -- alarms active.

```
currentAlarmTable OBJECT-TYPE
  SYNTAX SEQUENCE OF CurrentAlarmEntry
  MAX-ACCESS not-accessible
  STATUS current
  DESCRIPTION
    "A table that contains information about NE parameter properties
    that have alarms currently active."
  ::= { propertyIdent 2 }
```

```
currentAlarmEntry OBJECT-TYPE
  SYNTAX CurrentAlarmEntry
  MAX-ACCESS not-accessible
  STATUS current
  DESCRIPTION
    "A list of information about each property with an alarm that is
    currently active."
```

The OID suffix for an entry in this table is constructed by appending the length of currentAlarmOID and then the components of currentAlarmOID to identify an instance.

The first two components of currentAlarmOID will generally be 1.3 which will be encoded separately as 1 and 3, not as a single value of 43 (decimal). When currentAlarmOID identifies a scalar object, it is expected that the final suffix of .0 will be included.

If currentAlarmOID was 1.3.5.0, then 4.1.3.5.0 are the resulting components of the OID suffix."

```
INDEX { currentAlarmOID }  
::= { currentAlarmTable 1 }
```

```
CurrentAlarmEntry ::= SEQUENCE {  
    currentAlarmOID  
        OBJECT IDENTIFIER,  
    currentAlarmAlarmState  
        INTEGER,  
    currentAlarmAlarmValue  
        Integer32  
}
```

```
currentAlarmOID OBJECT-TYPE  
SYNTAX OBJECT IDENTIFIER  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION
```

"This is the OID of the alarmed object whose current value makes an alarm active.

In the case of the alarmed analog object, the value of this object is equal to the value of the index parameterOID of the propertyTable.

In the case of the alarmed discrete object, the value of this object is equal to the value of the index discreteParameterOID of the discretePropertyTable.

Please note that in the case of the alarmed discrete object, the value of this object is not equal to the identity part of the OID of the instances in the corresponding row of the discretePropertyTable.

Example 1. Object commonInternalTemperature (HMS024).

If object commonInternalTemperature has a major alarm for a HHHI threshold value of 100 degrees Celsius defined in the propertyTable, and this alarm occurs then:

- (1) 'currentAlarmOID.12.commonInternalTemperature.0'  
instance will have the value 'commonInternalTemperature.0';
- (2) 'currentAlarmAlarmState.12.commonInternalTemperature.0'  
instance will have the value caasHHHI(2);
- (3) 'currentAlarmAlarmValue.12.commonInternalTemperature.0'  
instance will have the value 100.

Notice the presence of 12 in the OIDs of the instances above. Number 12 is the length of the 'commonInternalTemperature.0' OID, which is '1.3.6.1.4.1.5591.1.3.1.13.0'.

Example 2. Object fnOpticalReceiverABSwitchState (HMS025).

If object 'fnOpticalReceiverABSwitchState.1', which is the first instance of A/B switch in a fiber node, has a major alarm for a value pathB(2) defined in the discretePropertyTable, and this alarm occurs then:

- (1) 'currentAlarmOID.13.fnOpticalReceiverABSwitchState.1' instance will have the value 'fnOpticalReceiverABSwitchState.1';
- (2) 'currentAlarmAlarmState.13.fnOpticalReceiverABSwitchState.1' instance will have the value caasDiscreteMajor(6);
- (3) 'currentAlarmAlarmValue.13.fnOpticalReceiverABSwitchState.1' instance will have the value pathB(2).

Notice the presence of 13 in the OIDs of the instances above. Number 13 is the length of the 'fnOpticalReceiverABSwitchState.1' OID, which is '1.3.6.1.4.1.5591.1.5.13.1.4.1'.

Example 3. Object heCommonTemperature (HMS111).

If object 'commonInternalTemperature.1' has a major alarm for a HIHI threshold value of 60 degrees Celsius defined in the propertyTable, and this alarm occurs then:

- (1) 'currentAlarmOID.18.heCommonTemperature.1' instance will have the value 'heCommonTemperature.1';
- (2) 'currentAlarmAlarmState.18.heCommonTemperature.1' instance will have the value caasHIHI(2);
- (3) 'currentAlarmAlarmValue.18.heCommonTemperature.1' instance will have the value 600.

Notice the presence of 18 in the OIDs of the instances above. Number 18 is the length of the 'heCommonTemperature.1' OID, which is '1.3.6.1.4.1.5591.1.11.2.1.1.1.1.1.2.1'.

Example 4. Object heOpTxLaserOutputStatus (HMS112).

If object 'heOpTxLaserOutputStatus.1.2', which is the second laser

instance in the first instance of the headend optical transmitter, has a major alarm for a value off(1) defined in the discretePropertyTable, and this alarm occurs then:

- (1) 'currentAlarmOID.18.heOpTxLaserOutputStatus.1.2' instance will have the value 'heOpTxLaserOutputStatus.1.2';
- (2) 'currentAlarmAlarmState.18.heOpTxLaserOutputStatus.1.2' instance will have the value caasDiscreteMajor(6);
- (3) 'currentAlarmAlarmValue.18.heOpTxLaserOutputStatus.1.2' instance will have the value off(1).

Notice the presence of 18 in the OIDs of the instances above. Number 18 is the length of the 'heOpTxLaserOutputStatus.2' OID, which is '1.3.6.1.4.1.5591.1.11.1.1.1.1.3.1.8.1.2'.

::= { currentAlarmEntry 1 }

currentAlarmAlarmState OBJECT-TYPE

SYNTAX INTEGER {  
caasHIHI (2),  
caasHI (3),  
caasLO (4),  
caasLOLO (5),  
caasDiscreteMajor (6),  
caasDiscreteMinor (7)

}  
MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The object contains the current alarm state of the associated property entry."

::= { currentAlarmEntry 2 }

currentAlarmAlarmValue OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only  
 STATUS current  
 DESCRIPTION  
 "Value that caused this alarm.

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller) values."

::= { currentAlarmEntry 3 }

-- \* The discrete property table contains information that must be supported by all OSP  
 -- \* network elements, including transponders, line monitors, amplifiers  
 -- \* fiber nodes, batteries, .... It provides the possibility to define  
 -- \* and enable alarms for the network element.

-- \* Usage of this table:

-- \* The table has a fixed number of rows (defined at design-time). All variables in  
 -- \* the HMS MIBs that need properties with discrete alarms have at least one entry  
 -- \* in the table defined in this table. Only discrete variables can have an one entry.

-- \* The table has 2 indices. The first one is the OID of the parameter for which  
 -- \* a property has to be set. The second one is discreteAlarmValue. The column discreteAlarmState  
 -- \* has to be used.

-- \* Example:

- \* - Variable: psInverterStatus.1
- \* - Alarms: testStarted(4), testFailed(5)
- \* - Two rows will be present in the table:

```
-- * +=====+=====+=====+=====+
-- * |discreteParameterOID |discreteAlarmValue |discreteAlarmEnable|discreteAlarmState |
-- * +=====+=====+=====+=====+
-- * |psInverterStatus.1 | 4 | 01h/02h | 01h/06h |
-- * |psInverterStatus.1 | 5 | 01h/02h | 01h/06h |
-- * +=====+=====+=====+=====+
```

-- \* - To enable or disable certain alarms, the correct bits have to be set in alarmEnable.

discretePropertyTable OBJECT-TYPE  
SYNTAX SEQUENCE OF DiscretePropertyEntry  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION  
"A table that contains information about NE parameter properties."  
 ::= { propertyIdent 3 }

discretePropertyEntry OBJECT-TYPE  
SYNTAX DiscretePropertyEntry  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION  
"A list of information about each property.

The OID suffix for an entry in this table is constructed by appending the length of discreteParameterOID and then the components of discreteParameterOID to identify an instance.

The first two components of discreteParameterOID will generally be 1.3 which will be encoded separately as 1 and 3, not as a single value of 43 (decimal). When discreteParameterOID identifies a scalar object, it is expected that the final suffix of .0 will be included.

If discreteParameterOID was 1.3.5.0, then 4.1.3.5.0 are the resulting components of the OID suffix."

INDEX { discreteParameterOID,  
discreteAlarmValue }  
 ::= { discretePropertyTable 1 }

DiscretePropertyEntry ::= SEQUENCE {  
discreteParameterOID  
OBJECT IDENTIFIER,  
discreteAlarmValue  
Integer32,  
discreteAlarmEnable

```
    INTEGER,  
    discreteAlarmState  
    INTEGER  
}
```

```
discreteParameterOID OBJECT-TYPE  
SYNTAX OBJECT IDENTIFIER  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "First index into discretePropertyTable. This is the OID of the parameter  
    whose property is being accessed.
```

Example: OID of psTamper for power supply 1 is 1.3.6.1.4.1.5591.1.4.2.1.27.1"  
::= { discretePropertyEntry 1 }

```
discreteAlarmValue OBJECT-TYPE  
SYNTAX Integer32 (0..2147483647) -- has to be a non-negative number  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "Second index into the discretePropertyTable.  
    When the parameter, specified by discreteParameterOID has this value,  
    an alarm will occur.
```

Some devices only require 16 bit integer or smaller and therefore only support 16 bit (or smaller)  
values."  
::= { discretePropertyEntry 2 }

```
discreteAlarmEnable OBJECT-TYPE  
SYNTAX INTEGER {  
    disable (1),  
    enableMajor (2),  
    enableMinor (3)  
}  
MAX-ACCESS read-write  
STATUS current
```

#### DESCRIPTION

"When set to enable(2 or 3), alarm processing for this property is enabled. When set to disable(1), alarm processing for this property is disabled. No entries into the alarmLogTable nor traps are permitted due to this property when in the disable(1) state. The default state for this object is disable(1).

This object should be kept in NV memory"

::= { discretePropertyEntry 3 }

#### discreteAlarmState OBJECT-TYPE

SYNTAX INTEGER {

dasNominal (1),

dasDiscreteMajor (6),

dasDiscreteMinor (7)

}

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"This object contains the current alarm state for this discrete property entry."

::= { discretePropertyEntry 4 }

#### propertyMIBCompliance MODULE-COMPLIANCE

STATUS current

#### DESCRIPTION

"The compliance statement for HMS entities which implement the SCTE HMS Property MIB."

#### MODULE

MANDATORY-GROUPS { analogAlarmsGroup,

currentAlarmsGroup,

discreteAlarmsGroup }

::= { propertyMIBCompliances 1 }

#### analogAlarmsGroup OBJECT-GROUP

OBJECTS {

parameterOID,

alarmEnable,

currentAlarmState,

analogAlarmHIHI,

```

    analogAlarmHI,
    analogAlarmLO,
    analogAlarmLOLO,
    analogAlarmDeadband
}
STATUS    current
DESCRIPTION
    "The analog alarms group defines objects which represent alarm information
    for alarmable analog variables in an optical module."
::= { propertyMIBGroups 1 }

discreteAlarmsGroup OBJECT-GROUP
OBJECTS { discreteParameterOID,
          discreteAlarmValue,
          discreteAlarmEnable,
          discreteAlarmState }
STATUS    current
DESCRIPTION
    "The discrete alarms group defines objects which represent alarm information
    for alarmable discrete variables in an optical module."
::= { propertyMIBGroups 2 }

currentAlarmsGroup OBJECT-GROUP
OBJECTS { currentAlarmAlarmState,
          currentAlarmAlarmValue,
          currentAlarmOID }
STATUS    current
DESCRIPTION
    "The current alarms group defines objects which represent a list of
    active alarms present in an optical module."
::= { propertyMIBGroups 3 }

END

```