

FDDI Management Information Base

Status of this Memo

This memo is an extension to the SNMP MIB. This RFC specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Abstract	1
2. The Network Management Framework.....	1
3. Objects	2
3.1 Format of Definitions	2
4. Overview	3
4.1 Textual Conventions	3
5. Object Definitions	4
5.1 The SMT Group	5
5.2 The MAC Group	15
5.3 The PATH Group	27
5.4 The PORT Group	27
5.5 The ATTACHMENT Group	38
5.6 The Chip Set Group	42
6. Acknowledgements	43
7. References	45
Security Considerations.....	46
Author's Address.....	46

1. Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP-based internets. In particular, it defines objects for managing devices which implement the FDDI.

2. The Network Management Framework

The Internet-standard Network Management Framework consists of three components. They are:

RFC 1155 which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management. RFC 1212 defines a more concise description mechanism, which is wholly consistent with the SMI.

RFC 1156 which defines MIB-I, the core set of managed objects for the Internet suite of protocols. RFC 1213, defines MIB-II, an evolution of MIB-I based on implementation experience and new operational requirements.

RFC 1157 which defines the SNMP, the protocol used for network access to managed objects.

The Framework permits new objects to be defined for the purpose of experimentation and evaluation.

3. Objects

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the subset of Abstract Syntax Notation One (ASN.1) [5] defined in the SMI. In particular, each object has a name, a syntax, and an encoding. The name is an object identifier, an administratively assigned name, which specifies an object type. The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the OBJECT DESCRIPTOR, to also refer to the object type.

The syntax of an object type defines the abstract data structure corresponding to that object type. The ASN.1 language is used for this purpose. However, the SMI [1] purposely restricts the ASN.1 constructs which may be used. These restrictions are explicitly made for simplicity.

The encoding of an object type is simply how that object type is represented using the object type's syntax. Implicitly tied to the notion of an object type's syntax and encoding is how the object type is represented when being transmitted on the network.

The SMI specifies the use of the basic encoding rules of ASN.1 [6], subject to the additional requirements imposed by the SNMP.

3.1. Format of Definitions

Section 5 contains contains the specification of all object types contained in this MIB module. The object types are defined using the conventions defined in the SMI, as amended by the extensions

specified in [7].

4. Overview

This document defines the managed objects for FDDI devices which are to be accessible via the Simple Network Management Protocol (SNMP). At present, this applies to these values of the ifType variable in the Internet-standard MIB:

```
fddi(15)
```

For these interfaces, the value of the ifSpecific variable in the MIB-II [4] has the OBJECT IDENTIFIER value:

```
fddi      OBJECT IDENTIFIER ::= { transmission 15 }
```

The definitions of the objects presented here draws heavily from related work in the ANSI X3T9.5 committee and the SMT subcommittee of that committee [8]. In fact, the definitions of the managed objects in this document are, to the maximum extent possible, identical to those identified by the ANSI committee. The semantics of each managed object should be the same with syntactic changes made as necessary to recast the objects in terms of the Internet-standard SMI and MIB so as to be compatible with the SNMP. Examples of these syntactic changes include remapping booleans to enumerated integers, remapping bit strings to octet strings, and the like. In addition, the naming of the objects was changed to achieve compatibility.

These minimal syntactic changes with no semantic changes should allow implementations of SNMP manageable FDDI systems to share instrumentation with other network management schemes and thereby minimize implementation cost. In addition, the translation of information conveyed by managed objects from one network management scheme to another is eased by these shared definitions.

Only the essential variables, as indicated by their mandatory status in the ANSI specification were retained in this document. The importance of variables which have an optional status in the ANSI specification were perceived as being less widely accepted.

4.1. Textual Conventions

Several new datatypes are introduced as a textual convention in this MIB document. These textual conventions enhance the readability of the document and ease comparisons with its ANSI counterpart. It should be noted that the introduction of the following textual conventions has no effect on either the syntax nor the semantics of any managed objects. The use of these is merely an artifact of the

explanatory method used. Objects defined in terms of one of these methods are always encoded by means of the rules that define the primitive type. Hence, no changes to the SMI or the SNMP are necessary to accommodate these textual conventions which are adopted merely for the convenience of readers and writers in pursuit of the elusive goal of clear, concise, and unambiguous MIB documents.

5. Object Definitions

```
RFC1285-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    Counter
        FROM RFC1155-SMI
    transmission
        FROM RFC1213-MIB
    OBJECT-TYPE
        FROM RFC-1212;
```

```
-- This MIB module uses the extended OBJECT-TYPE macro as
-- defined in [7].
```

```
-- this is the FDDI MIB module
```

```
fddi    OBJECT IDENTIFIER ::= { transmission 15 }
```

```
-- textual conventions
```

```
FddiTime ::= INTEGER (0..2147483647)
```

```
-- This data type specifies octet units of 80 nanoseconds as
-- an integer value. It is used for Path Latency and
-- Synchronous Bandwidth values. The encoding is normal
-- integer representation (not twos complement).
```

```
FddiResourceId ::= INTEGER (0..65535)
```

```
-- This data type is used to refer to an instance of a MAC,
-- PORT, PATH, or ATTACHMENT Resource ID. Indexing begins
-- at 1. Zero is used to indicate the absence of a resource.
```

```
FddiSMTStationIdType ::= OCTET STRING (SIZE (8))
```

```
-- The unique identifier for the FDDI station. This is a
-- string of 8 octets, represented as
```

```
--          X' yy yy xx xx xx xx xx xx'
-- with the low order 6 octet (xx) from a unique IEEE
-- assigned address. The high order two bits of the IEEE
-- address, the group address bit and the administration bit
```

```
-- (Universal/Local) bit should both be zero.  The first two
-- octets, the yy octets, are implementor-defined.
--
-- The representation of the address portion of the station id
-- is in the IEEE (ANSI/IEEE P802.1A) canonical notation for
-- 48 bit addresses.  The canonical form is a 6-octet string
-- where the first octet contains the first 8 bits of the
-- address, with the I/G(Individual/Group) address bit as the
-- least significant bit and the U/L (Universal/Local) bit
-- as the next more significant bit, and so on.  Note that
-- addresses in the ANSI FDDI standard SMT frames are
-- represented in FDDI MAC order.
```

```
FddiMACLongAddressType ::= OCTET STRING (SIZE (6))
-- The representation of long MAC addresses as management
-- values is in the IEEE (ANSI/IEEE P802.1A) canonical
-- notation for 48 bit addresses.  The canonical form is a
-- 6-octet string where the first octet contains the first 8
-- bits of the address, with the I/G (Individual/Group)
-- address bit as the least significant bit and the U/L
-- (Universal/Local) bit as the next more significant bit,
-- and so on.  Note that the addresses in the SMT frames are
-- represented in FDDI MAC order.
```

```
-- groups in the FDDI MIB module
```

```
snmpFddiSMT          OBJECT IDENTIFIER ::= { fddi 1 }

snmpFddiMAC          OBJECT IDENTIFIER ::= { fddi 2 }

snmpFddiPATH         OBJECT IDENTIFIER ::= { fddi 3 }

snmpFddiPORT         OBJECT IDENTIFIER ::= { fddi 4 }

snmpFddiATTACHMENT   OBJECT IDENTIFIER ::= { fddi 5 }

snmpFddiChipSets     OBJECT IDENTIFIER ::= { fddi 6 }
```

```
-- the SMT group
-- Implementation of the SMT group is mandatory for all
-- systems which implement manageable FDDI subsystems.
```

```
snmpFddiSMTNumber OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
```

```

        "The number of SMT implementations (regardless of
        their current state) on this network management
        application entity. The value for this variable
        must remain constant at least from one re-
        initialization of the entity's network management
        system to the next re-initialization."
 ::= { snmpFddiSMT 1 }

-- the SMT table

snmpFddiSMTTable OBJECT-TYPE
    SYNTAX SEQUENCE OF SnmpFddiSMTEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "A list of SMT entries. The number of entries is
        given by the value of snmpFddiSMTNumber."
    ::= { snmpFddiSMT 2 }

snmpFddiSMTEntry OBJECT-TYPE
    SYNTAX SnmpFddiSMTEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
        "An SMT entry containing information common to a
        given SMT."
    INDEX { snmpFddiSMTIndex }
    ::= { snmpFddiSMTTable 1 }

SnmpFddiSMTEntry ::=
    SEQUENCE {
        snmpFddiSMTIndex
            INTEGER,
        snmpFddiSMTStationId
            FddiSMTStationIdType,
        snmpFddiSMTOpVersionId
            INTEGER,
        snmpFddiSMTHiVersionId
            INTEGER,
        snmpFddiSMTLoVersionId
            INTEGER,
        snmpFddiSMTMACCt
            INTEGER,
        snmpFddiSMTNonMasterCt
            INTEGER,
        snmpFddiSMTMasterCt
            INTEGER,

```

```

    snmpFddiSMTPathsAvailable
        INTEGER,
    snmpFddiSMTConfigCapabilities
        INTEGER,
    snmpFddiSMTConfigPolicy
        INTEGER,
    snmpFddiSMTConnectionPolicy
        INTEGER,
    snmpFddiSMTTNotify
        INTEGER,
    snmpFddiSMTStatusReporting
        INTEGER,
    snmpFddiSMTECMState
        INTEGER,
    snmpFddiSMTCFState
        INTEGER,
    snmpFddiSMTHoldState
        INTEGER,
    snmpFddiSMTRemoteDisconnectFlag
        INTEGER,
    snmpFddiSMTStationAction
        INTEGER
}

snmpFddiSMTIndex OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A unique value for each SMT.  Its value ranges
        between 1 and the value of snmpFddiSMTNumber.  The
        value for each SMT must remain constant at least
        from one re-initialization of the entity's network
        management system to the next re-initialization."
    ::= { snmpFddiSMTEntry 1 }

snmpFddiSMTStationId OBJECT-TYPE
    SYNTAX  FddiSMTStationIdType -- OCTET STRING (SIZE (8))
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "Uniquely identifies an FDDI station."
    REFERENCE
        "ANSI { fddiSMT 11 }"
    ::= { snmpFddiSMTEntry 2 }

```

```
snmpFddiSMTOpVersionId OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The version that this station is using for its
        operation (refer to ANSI 7.1.2.2)."
```

```
REFERENCE
    "ANSI { fddiSMT 13 }"
 ::= { snmpFddiSMTEntry 3 }
```

```
snmpFddiSMTHiVersionId OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The highest version of SMT that this station
        supports (refer to ANSI 7.1.2.2)."
```

```
REFERENCE
    "ANSI { fddiSMT 14 }"
 ::= { snmpFddiSMTEntry 4 }
```

```
snmpFddiSMTLoVersionId OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The lowest version of SMT that this station
        supports (refer to ANSI 7.1.2.2)."
```

```
REFERENCE
    "ANSI { fddiSMT 15 }"
 ::= { snmpFddiSMTEntry 5 }
```

```
snmpFddiSMTMACCt OBJECT-TYPE
    SYNTAX  INTEGER (0..255)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The number of MACs in the station or
        concentrator."
```

```
REFERENCE
    "ANSI { fddiSMT 21 }"
 ::= { snmpFddiSMTEntry 6 }
```

```
snmpFddiSMTNonMasterCt OBJECT-TYPE
    SYNTAX  INTEGER (0..2)
    ACCESS  read-only
    STATUS  mandatory
```


DESCRIPTION

"The number of Non Master PORTs (A, B, or S PORTs) in the station or concentrator."

REFERENCE

"ANSI { fddiSMT 22 }"
 ::= { snmpFddiSMTEntry 7 }

snmpFddiSMTMasterCt OBJECT-TYPE

SYNTAX INTEGER (0..255)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The number of Master PORTs in a node. If the node is not a concentrator, the value is zero."

REFERENCE

"ANSI { fddiSMT 23 }"
 ::= { snmpFddiSMTEntry 8 }

snmpFddiSMTPathsAvailable OBJECT-TYPE

SYNTAX INTEGER (0..7)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A value that indicates the PATH types available in the station.

The value is a sum. This value initially takes the value zero, then for each type of PATH that this node has available, 2 raised to a power is added to the sum. The powers are according to the following table:

Path	Power
Primary	0
Secondary	1
Local	2

For example, a station having Primary and Local PATHs available would have a value of 5 ($2^{**0} + 2^{**2}$)."

REFERENCE

"ANSI { fddiSMT 24 }"
 ::= { snmpFddiSMTEntry 9 }

snmpFddiSMTConfigCapabilities OBJECT-TYPE

SYNTAX INTEGER (0..3)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A value that indicates capabilities that are present in the node. If 'holdAvailable' is present, this indicates support of the optional Hold Function (refer to ANSI SMT 9.4.3.2). If 'CF-Wrap-AB' is present, this indicates that the WRAP_AB state is forced.

The value is a sum. This value initially takes the value zero, then for each of the configuration policies currently enforced on the node, 2 raised to a power is added to the sum. The powers are according to the following table:

Policy	Power
holdAvailable	0
CF-Wrap-AB	1 "

REFERENCE

"ANSI { fddiSMT 25 }"
 ::= { snmpFddiSMTEntry 10 }

snmpFddiSMTConfigPolicy OBJECT-TYPE

SYNTAX INTEGER (0..3)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"A value that indicates the configuration policies currently enforced in the node (refer to ANSI SMT 9.4.3.2). The 'configurationHold' policy refers to the Hold flag, and should not be present only if the Hold function is supported. The 'CF-Wrap-AB' policy refers to the CF_Wrap_AB flag.

The value is a sum. This value initially takes the value zero, then for each of the configuration policies currently enforced on the node, 2 raised to a power is added to the sum. The powers are according to the following table:

Policy	Power
configurationHold	0
CF-Wrap-AB	1 "

REFERENCE

"ANSI { fddiSMT 26 }"
 ::= { snmpFddiSMTEntry 11 }

snmpFddiSMTConnectionPolicy OBJECT-TYPE

SYNTAX INTEGER (0..65535)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"A value that indicates the connection policies enforced at the station. A station sets the corresponding policy for each of the connection types that it rejects. The letter designations, X and Y, in the 'rejectX-Y' names have the following significance: X represents the PC-Type of the local PORT and Y represents a PC-Neighbor in the evaluation of Connection-Policy (PC-Type, PC-Neighbor) that is done to determine the setting of T-Val(3) in the PC-Signaling sequence (refer to ANSI Section 9.6.3).

The value is a sum. This value initially takes the value zero, then for each of the connection policies currently enforced on the node, 2 raised to a power is added to the sum. The powers are according to the following table:

Policy	Power
rejectA-A	0
rejectA-B	1
rejectA-S	2
rejectA-M	3
rejectB-A	4
rejectB-B	5
rejectB-S	6
rejectB-M	7
rejectS-A	8
rejectS-B	9
rejectS-S	10
rejectS-M	11
rejectM-A	12
rejectM-B	13
rejectM-S	14
rejectM-M	15

Implementors should note that the polarity of these bits is different in different places in an SMT system. Implementors should take appropriate care."

REFERENCE

"ANSI { fddiSMT 27 }"
 ::= { snmpFddiSMTEntry 12 }

snmpFddiSMTTNotify OBJECT-TYPE

SYNTAX INTEGER (2..30)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The timer used in the Neighbor Notification protocol, reported in seconds and ranging from 2 to 30 seconds (refer to ANSI SMT 8.3.1)."

REFERENCE

"ANSI { fddiSMT 29 }"

::= { snmpFddiSMTEntry 13 }

snmpFddiSMTStatusReporting OBJECT-TYPE

SYNTAX INTEGER { true(1), false(2) }

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Indicates whether the node implements the Status Reporting Protocol. This object is included for compatibility with products that were designed prior to the adoption of this standard."

REFERENCE

"ANSI { fddiSMT 30 }"

::= { snmpFddiSMTEntry 14 }

snmpFddiSMTTECMState OBJECT-TYPE

```
SYNTAX  INTEGER {
    ec0(1), -- Out
    ec1(2), -- In
    ec2(3), -- Trace
    ec3(4), -- Leave
    ec4(5), -- Path_Test
    ec5(6), -- Insert
    ec6(7), -- Check
    ec7(8)  -- Deinsert
}
```

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Indicates the current state of the ECM state machine (refer to ANSI SMT 9.5.2)."

REFERENCE

"ANSI { fddiSMT 41 }"

::= { snmpFddiSMTEntry 15 }

snmpFddiSMTCFState OBJECT-TYPE

```
SYNTAX  INTEGER {
    cf0(1), -- Isolated
```

```

        cf1(2), -- Wrap_S
        cf2(3), -- Wrap_A
        cf3(4), -- Wrap_B
        cf4(5), -- Wrap_AB
        cf5(6)  -- Thru
    }
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The attachment configuration for the station or
    concentrator (refer to ANSI SMT 9.7.4.3)."
```

REFERENCE

```

    "ANSI { fddiSMT 42 }"
::= { snmpFddiSMTEntry 16 }
```

snmpFddiSMTHoldState OBJECT-TYPE

```

SYNTAX  INTEGER {
    not-implemented(1), -- holding not implemented
    not-holding(2),
    holding-prm(3),      -- holding on primary
    holding-sec(4)       -- holding on secondary
}
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "This value indicates the current state of the
    Hold function. The values are determined as
    follows: 'holding-prm' is set if the primary ring
    is operational and the Recovery Enable Flag is
    clear (NOT NO_Flag(primary) AND NOT RE_Flag). is
    set if the secondary ring is operational and the
    Recovery Enable Flag is clear (NOT
    NO_Flag(secondary) AND NOT RE_Flag). Ref 9.4.3.
    and 10.3.1. the primary or secondary, i.e., the
    Recovery Enable, RE_Flag, is set."
```

REFERENCE

```

    "ANSI { fddiSMT 43 }"
::= { snmpFddiSMTEntry 17 }
```

snmpFddiSMTRemoteDisconnectFlag OBJECT-TYPE

```

SYNTAX  INTEGER { true(1), false(2) }
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "A flag indicating that the station was remotely
    disconnected from the network. A station requires
    a Connect Action (SM_CM_CONNECT.request (Connect))
    to rejoin and clear the flag (refer to ANSI
```

6.4.5.2)."

REFERENCE

"ANSI { fddiSMT 44 }"
 ::= { snmpFddiSMTEntry 18 }

snmpFddiSMTStationAction OBJECT-TYPE

SYNTAX INTEGER {
 other(1), -- none of the following
 connect(2),
 disconnect(3),
 path-Test(4),
 self-Test(5)
 }

ACCESS read-write

STATUS mandatory

DESCRIPTION

"This object, when read, always returns a value of other(1). The behavior of setting this variable to each of the acceptable values is as follows:

Other: Results in a badValue error.

Connect: Generates an SM_CM_Connect.request(connect) signal to CMT indicating that the ECM State machine is to begin a connection sequence. The fddiSMTRemoteDisconnectFlag is cleared on the setting of this variable to 1. See ANSI Ref 9.3.1.1.

Disconnect: Generates an SM_CM_Connect.request(disconnect) signal to ECM and sets the fddiSMTRemoteDisconnectFlag. See ANSI Ref 9.3.1.1.

Path-Test: Initiates a station path test. The Path_Test variable (See ANSI Ref. 9.4.1) is set to Testing. The results of this action are not specified in this standard.

Self-Test: Initiates a station self test. The results of this action are not specified in this standard.

Attempts to set this object to all other values results in a badValue error. Agents may elect to return a badValue error on attempts to set this variable to path-Test(4) or self-Test(5)."

REFERENCE

```
"ANSI { fddiSMT 60 }"
 ::= { snmpFddiSMTEntry 19 }
```

```
-- the MAC group
-- Implementation of the MAC Group is mandatory for all
-- systems which implement manageable FDDI subsystems.
```

snmpFddiMACNumber OBJECT-TYPE

```
SYNTAX  INTEGER (0..65535)
```

```
ACCESS  read-only
```

```
STATUS  mandatory
```

DESCRIPTION

```
"The total number of MAC implementations (across
all SMTs) on this network management application
entity. The value for this variable must remain
constant at least from one re-initialization of
the entity's network management system to the next
re-initialization."
```

```
::= { snmpFddiMAC 1 }
```

```
-- the MAC table
```

snmpFddiMACTable OBJECT-TYPE

```
SYNTAX  SEQUENCE OF SnmpFddiMACEntry
```

```
ACCESS  not-accessible
```

```
STATUS  mandatory
```

DESCRIPTION

```
"A list of MAC entries. The number of entries is
given by the value of snmpFddiMACNumber."
```

```
::= { snmpFddiMAC 2 }
```

snmpFddiMACEntry OBJECT-TYPE

```
SYNTAX  SnmpFddiMACEntry
```

```
ACCESS  not-accessible
```

```
STATUS  mandatory
```

DESCRIPTION

```
"A MAC entry containing information common to a
given MAC."
```

```
INDEX   { snmpFddiMACSMTIndex, snmpFddiMACIndex }
```

```
::= { snmpFddiMACTable 1 }
```

SnmpFddiMACEntry ::=

```
SEQUENCE {
```

```
    snmpFddiMACSMTIndex
```

```
    INTEGER,
```

```
snmpFddiMACIndex
    INTEGER,
snmpFddiMACFrameStatusCapabilities
    INTEGER,
snmpFddiMACTMaxGreatestLowerBound
    FddiTime,
snmpFddiMACTVXGreatestLowerBound
    FddiTime,
snmpFddiMACPathsAvailable
    INTEGER,
snmpFddiMACCurrentPath
    INTEGER,
snmpFddiMACUpstreamNbr
    FddiMACLongAddressType,
snmpFddiMACOldUpstreamNbr
    FddiMACLongAddressType,
snmpFddiMACDupAddrTest
    INTEGER,
snmpFddiMACPathsRequested
    INTEGER,
snmpFddiMACDownstreamPORTType
    INTEGER,
snmpFddiMACSMTAddress
    FddiMACLongAddressType,
snmpFddiMACTReq
    FddiTime,
snmpFddiMACTNeg
    FddiTime,
snmpFddiMACTMax
    FddiTime,
snmpFddiMACTvxValue
    FddiTime,
snmpFddiMACTMin
    FddiTime,
snmpFddiMACCurrentFrameStatus
    INTEGER,
snmpFddiMACFrameCts
    Counter,
snmpFddiMACErrorCts
    Counter,
snmpFddiMACLostCts
    Counter,
snmpFddiMACFrameErrorThreshold
    INTEGER,
snmpFddiMACFrameErrorRatio
    INTEGER,
snmpFddiMACRMTState
    INTEGER,
```



```
        snmpFddiMACDaFlag
            INTEGER,
        snmpFddiMACUnaDaFlag
            INTEGER,
        snmpFddiMACFrameCondition
            INTEGER,
        snmpFddiMACChipSet
            OBJECT IDENTIFIER,
        snmpFddiMACAction
            INTEGER
    }

snmpFddiMACSMTIndex OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of the SMT index associated with this
        MAC."
    ::= { snmpFddiMACEntry 1 }

snmpFddiMACIndex OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A unique value for each MAC on the managed
        entity. The MAC identified by a particular value
        of this index is that identified by the same value
        of an ifIndex object instance. That is, if a MAC
        is associated with the interface whose value of
        ifIndex in the Internet-Standard MIB is equal to
        5, then the value of snmpFddiMACIndex shall also
        equal 5. The value for each MAC must remain
        constant at least from one re-initialization of
        the entity's network management system to the next
        re-initialization."
    ::= { snmpFddiMACEntry 2 }

snmpFddiMACFrameStatusCapabilities OBJECT-TYPE
    SYNTAX  INTEGER (0..1799)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A value that indicates the MAC's bridge and end-
        station capabilities for operating in a bridged
        FDDI network."
```

The value is a sum. This value initially takes the value zero, then for each capability present, 2 raised to a power is added to the sum. The powers are according to the following table:

Capability	Power
FSC-Type0	0
-- MAC repeats A/C indicators as received on copying with the intent to forward.	
FSC-Type1	1
-- MAC sets C but not A on copying for forwarding.	
FSC-Type2	2
-- MAC resets C and sets A on C set and A reset if the frame is not copied and the frame was addressed to this MAC	
FSC-Type0-programmable	8
-- Type0 capability is programmable	
FSC-Type1-programmable	9
-- Type1 capability is programmable	
FSC-Type2-programmable	10
-- Type2 capability is programmable	

"

REFERENCE

"ANSI { fddiMAC 11 }"
 ::= { snmpFddiMACEntry 3 }

snmpFddiMACTMaxGreatestLowerBound OBJECT-TYPE

SYNTAX FddiTime
 ACCESS read-write
 STATUS mandatory
 DESCRIPTION

"The greatest lower bound of T_Max supported for this MAC."

REFERENCE

"ANSI { fddiMAC 13 }"
 ::= { snmpFddiMACEntry 4 }

snmpFddiMACTVXGreatestLowerBound OBJECT-TYPE

SYNTAX FddiTime
 ACCESS read-only
 STATUS mandatory

DESCRIPTION

"The greatest lower bound of TVX supported for this MAC."

REFERENCE

"ANSI { fddiMAC 14 }"
 ::= { snmpFddiMACEntry 5 }

snmpFddiMACPathsAvailable OBJECT-TYPE

SYNTAX INTEGER (0..7)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A value that indicates the PATH types available for this MAC."

The value is a sum. This value initially takes the value zero, then for each type of PATH that this MAC has available, 2 raised to a power is added to the sum. The powers are according to the following table:

Path	Power
Primary	0
Secondary	1
Local	2 "

REFERENCE

"ANSI { fddiMAC 22 }"
 ::= { snmpFddiMACEntry 6 }

snmpFddiMACCurrentPath OBJECT-TYPE

SYNTAX INTEGER {
 unknown(1),
 primary(2),
 secondary(4),
 local(8),
 isolated(16)
 }

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Indicates the association of the MAC with a station PATH."

REFERENCE

"ANSI { fddiMAC 23 }"
 ::= { snmpFddiMACEntry 7 }

snmpFddiMACUpstreamNbr OBJECT-TYPE

SYNTAX FddiMACLongAddressType -- OCTET STRING (SIZE (6))

```

ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The MAC's upstream neighbor's long individual MAC
    address. It may be determined by the Neighbor
    Information Frame protocol (refer to ANSI SMT
    7.2.1). The value shall be reported as '00 00 00
    00 00 00' if it is unknown."
REFERENCE
    "ANSI { fddiMAC 24 }"
::= { snmpFddiMACEntry 8 }

```

```

snmpFddiMACOldUpstreamNbr OBJECT-TYPE
SYNTAX    FddiMACLongAddressType -- OCTET STRING (SIZE (6))
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The previous value of the MAC's upstream
    neighbor's long individual MAC address. It may be
    determined by the Neighbor Information Frame
    protocol (refer to ANSI SMT 7.2.1). The value
    shall be reported as '00 00 00 00 00 00' if it is
    unknown."
REFERENCE
    "ANSI { fddiMAC 26 }"
::= { snmpFddiMACEntry 9 }

```

```

snmpFddiMACDupAddrTest OBJECT-TYPE
SYNTAX    INTEGER { none(1), pass(2), fail(3) }
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "The Duplicate Address Test flag, Dup_Addr_Test
    (refer to ANSI 8.3.1)."
REFERENCE
    "ANSI { fddiMAC 29 }"
::= { snmpFddiMACEntry 10 }

```

```

snmpFddiMACPathsRequested OBJECT-TYPE
SYNTAX    INTEGER
ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "A value that indicates PATH(s) desired for this
    MAC.

    The value is a sum which represents the individual
    PATHs that are desired. This value initially

```

takes the value zero, then for each type of PATH that this node is, 2 raised to a power is added to the sum. The powers are according to the following table:

Path	Power
Primary	0
Secondary	1
Local	2
Isolated	3

The precedence order is primary, secondary, local, and then isolated if multiple PATHs are desired are set."

REFERENCE

"ANSI { fddiMAC 32 }"
 ::= { snmpFddiMACEntry 11 }

snmpFddiMACDownstreamPORTType OBJECT-TYPE

SYNTAX INTEGER { a(1), b(2), s(3), m(4), unknown(5) }

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Indicates the PC-Type of the first port that is downstream of this MAC (the exit port)."

REFERENCE

"ANSI { fddiMAC 33 }"
 ::= { snmpFddiMACEntry 12 }

snmpFddiMACSMTAddress OBJECT-TYPE

SYNTAX FddiMACLongAddressType -- OCTET STRING (SIZE (6))

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The 48 bit individual address of the MAC used for SMT frames."

REFERENCE

"ANSI { fddiMAC 41 }"
 ::= { snmpFddiMACEntry 13 }

snmpFddiMACTReq OBJECT-TYPE

SYNTAX FddiTime

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The value of T-Req (refer to ANSI MAC 2.2.1 and ANSI MAC 7.3.5.2)."

REFERENCE

```
        "ANSI { fddiMAC 51 }"
 ::= { snmpFddiMACEntry 14 }

snmpFddiMACTNeg OBJECT-TYPE
    SYNTAX  FddiTime
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of T-Neg (refer to ANSI MAC 2.2.1 and
        ANSI MAC 7.3.5.2)."
```

```
REFERENCE
    "ANSI { fddiMAC 52 }"
 ::= { snmpFddiMACEntry 15 }

snmpFddiMACTMax OBJECT-TYPE
    SYNTAX  FddiTime
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of T-Max (refer to ANSI MAC 2.2.1 and
        ANSI MAC 7.3.5.2)."
```

```
REFERENCE
    "ANSI { fddiMAC 53 }"
 ::= { snmpFddiMACEntry 16 }

snmpFddiMACTvxValue OBJECT-TYPE
    SYNTAX  FddiTime
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of TvxFValue (refer to ANSI MAC 2.2.1
        and ANSI MAC 7.3.5.2)."
```

```
REFERENCE
    "ANSI { fddiMAC 54 }"
 ::= { snmpFddiMACEntry 17 }

snmpFddiMACTMin OBJECT-TYPE
    SYNTAX  FddiTime
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of T-Min (refer to ANSI MAC 2.2.1 and
        ANSI MAC 7.3.5.2)."
```

```
REFERENCE
    "ANSI { fddiMAC 55 }"
 ::= { snmpFddiMACEntry 18 }
```

snmpFddiMACCurrentFrameStatus OBJECT-TYPE

SYNTAX INTEGER (0..7)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"A value that indicates the MAC's operational frame status setting functionality.

The value is a sum. This value initially takes the value zero, then for each functionality present, 2 raised to a power is added to the sum. The powers are according to the following table:

Functionality	Power
FSC-Type0	0
-- MAC repeats A/C indicators as received	
FSC-Type1	1
-- MAC sets C but not A on copying for forwarding	
FSC-Type2	2
-- MAC resets C and sets A on C set and A reset if frame is not copied	

"

REFERENCE

"ANSI { fddiMAC 63 }"

::= { snmpFddiMACEntry 19 }

snmpFddiMACFrameCts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Frame_Ct (refer to ANSI MAC 2.2.1)."

REFERENCE

"ANSI { fddiMAC 71 }"

::= { snmpFddiMACEntry 20 }

snmpFddiMACErrorCts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Error_Ct (refer to ANSI MAC 2.2.1)."

REFERENCE

"ANSI { fddiMAC 81 }"

::= { snmpFddiMACEntry 21 }

snmpFddiMACLostCts OBJECT-TYPE

SYNTAX Counter

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Lost_Ct (refer to ANSI MAC 2.2.1)."

REFERENCE

"ANSI { fddiMAC 82 }"

::= { snmpFddiMACEntry 22 }

snmpFddiMACFrameErrorThreshold OBJECT-TYPE

SYNTAX INTEGER (1..65535)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A threshold for determining when a MAC Condition report should be generated. The condition is true when the ratio, $((\text{delta snmpFddiMACLostCt} + \text{delta snmpFddiMACErrorCt}) / (\text{delta snmpFddiMACFrameCt} + \text{delta snmpFddiMACLostCt})) \times 2^{16}$, exceeds the threshold. It is used to determine when a station has an unacceptable frame error threshold. The sampling algorithm is implementation dependent. Any attempt to set this variable to a value of less than one shall result in a badValue error. Those who are familiar with the SNMP management framework will recognize that thresholds are not in keeping with the SNMP philosophy. However, this variable is supported by underlying SMT implementations already and maintaining this threshold should not pose an undue additional burden on SNMP agent implementors."

REFERENCE

"ANSI { fddiMAC 95 }"

::= { snmpFddiMACEntry 23 }

snmpFddiMACFrameErrorRatio OBJECT-TYPE

SYNTAX INTEGER (1..65535)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"This attribute is the actual ratio, $((\text{delta snmpFddiMACLostCt} + \text{delta snmpFddiMACErrorCt}) / (\text{delta snmpFddiMACFrameCt} + \text{delta snmpFddiMACLostCt})) \times 2^{16}$."

REFERENCE

"ANSI { fddiMAC 96 }"

::= { snmpFddiMACEntry 24 }

snmpFddiMACRMTState OBJECT-TYPE

```

SYNTAX  INTEGER {
                rm0(1), -- Isolated
                rm1(2), -- Non_Op
                rm2(3), -- Ring_Op
                rm3(4), -- Detect
                rm4(5), -- Non_Op_Dup
                rm5(6), -- Ring_Op_Dup
                rm6(7), -- Directed
                rm7(8)  -- Trace
            }
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "Indicates the current state of the Ring
    Management state machine (refer to ANSI Section
    10)."
```

REFERENCE

```

    "ANSI { fddiMAC 111 }"
 ::= { snmpFddiMACEntry 25 }
```

snmpFddiMACDaFlag OBJECT-TYPE

```

SYNTAX  INTEGER { true(1), false(2) }
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "The RMT flag Duplicate Address Flag, DA_Flag
    (refer to ANSI 10.3.1.2)."
```

REFERENCE

```

    "ANSI { fddiMAC 112 }"
 ::= { snmpFddiMACEntry 26 }
```

snmpFddiMACUnaDaFlag OBJECT-TYPE

```

SYNTAX  INTEGER { true(1), false(2) }
ACCESS  read-only
STATUS  mandatory
DESCRIPTION
    "A flag set when the upstream neighbor reports a
    duplicate address condition. Reset when the
    condition clears."
```

REFERENCE

```

    "ANSI { fddiMAC 113 }"
 ::= { snmpFddiMACEntry 27 }
```

snmpFddiMACFrameCondition OBJECT-TYPE

```

SYNTAX  INTEGER { true(1), false(2) }
ACCESS  read-only
STATUS  mandatory
```

DESCRIPTION

"Indicates the MAC Condition is active when set.
Cleared when the condition clears and on power
up."

REFERENCE

"ANSI { fddiMAC 114 }"
::= { snmpFddiMACEntry 28 }

snmpFddiMACChipSet OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"This object identifies the hardware chip(s) which is (are) principally responsible for the implementation of the MAC function. A few OBJECT IDENTIFIERS are identified elsewhere in this memo. For those The assignment of additional OBJECT IDENTIFIERS to various types of hardware chip sets is managed by the IANA. For example, vendors whose chip sets are not defined in this memo may request a number from the Internet Assigned Numbers Authority (IANA) which indicates the assignment of an enterprise specific subtree which, among other things, may be used to allocate OBJECT IDENTIFIER assignments for that enterprise's chip sets. Similarly, in the absence of an appropriately assigned OBJECT IDENTIFIER in this memo or in an enterprise specific subtree of a chip vendor, a board or system vendor can request a number for a subtree from the IANA and make an appropriate assignment. It is desired that, whenever possible, the same OBJECT IDENTIFIER be used for all chips of a given type. Consequently, the assignment made in this memo for a chip, if any, should be used in preference to any other assignment and the assignment made by the chip manufacturer, if any, should be used in preference to assignments made by users of those chips. If the hardware chip set is unknown, the object identifier

unknownChipSet OBJECT IDENTIFIER ::= { 0 0 }

is returned. Note that unknownChipSet is a syntactically valid object identifier, and any conformant implementation of ASN.1 and the BER must be able to generate and recognize this

```

        value."
 ::= { snmpFddiMACEntry 29 }

snmpFddiMACAction OBJECT-TYPE
    SYNTAX  INTEGER {
        other(1),          -- none of the following
        enableLLCService(2),
        disableLLCService(3),
        connectMAC(4),
        disconnectMAC(5)
    }
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "This object, when read, always returns a value of
        other(1).  The behavior of setting this variable
        to each of the acceptable values is as follows:

        Other:                      Results in a badValue
                                   error.

        enableLLCService:          enables MAC service to
                                   higher layers.

        disableLLCService:         disables MAC service to
                                   higher layers.

        connectMAC:                connect this MAC in
                                   station.

        disconnectMAC:            disconnect this MAC in
                                   station.

        Attempts to set this object to all other values
        results in a badValue error."
    REFERENCE
        "ANSI { fddiMAC 130 }"
 ::= { snmpFddiMACEntry 30 }

-- the PATH group

-- the PATH group is empty for now and shall remain so until
-- the ANSI community sorts out their PATH group

-- the PORT group
-- Implementation of the PORT group is mandatory for all

```

-- systems which implement manageable FDDI subsystems.

snmpFddiPORTNumber OBJECT-TYPE

SYNTAX INTEGER (0..65535)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The total number of PORT implementations (across all SMTs) on this network management application entity. The value for this variable must remain constant at least from one re-initialization of the entity's network management system to the next re-initialization."

::= { snmpFddiPORT 1 }

-- the PORT table

snmpFddiPORTTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnmpFddiPORTEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A list of PORT entries. The number of entries is given by the value of snmpFddiPORTNumber."

::= { snmpFddiPORT 2 }

snmpFddiPORTEntry OBJECT-TYPE

SYNTAX SnmpFddiPORTEntry

ACCESS not-accessible

STATUS mandatory

DESCRIPTION

"A PORT entry containing information common to a given PORT."

INDEX { snmpFddiPORTSMTIndex, snmpFddiPORTIndex }

::= { snmpFddiPORTTable 1 }

SnmpFddiPORTEntry ::=

SEQUENCE {

snmpFddiPORTSMTIndex

INTEGER,

snmpFddiPORTIndex

INTEGER,

snmpFddiPORTPCType

INTEGER,

snmpFddiPORTPCNeighbor

INTEGER,

snmpFddiPORTConnectionPolicies

```

        INTEGER,
snmpFddiPORTRemoteMACIndicated
        INTEGER,
snmpFddiPORTCEState
        INTEGER,
snmpFddiPORTPathsRequested
        INTEGER,
snmpFddiPORTMACPlacement
        FddiResourceId,
snmpFddiPORTAvailablePaths
        INTEGER,
snmpFddiPORTMACLoopTime
        FddiTime,
snmpFddiPORTTBMax
        FddiTime,
snmpFddiPORTBSFlag
        INTEGER,
snmpFddiPORTLCTFailCts
        Counter,
snmpFddiPORTLERestimate
        INTEGER,
snmpFddiPORTLEmRejectCts
        Counter,
snmpFddiPORTLEmCts
        Counter,
snmpFddiPORTLERcutoff
        INTEGER,
snmpFddiPORTLERAlarm
        INTEGER,
snmpFddiPORTConnectState
        INTEGER,
snmpFddiPORTPCMState
        INTEGER,
snmpFddiPORTPCWithhold
        INTEGER,
snmpFddiPORTLERCondition
        INTEGER,
snmpFddiPORTChipSet
        OBJECT IDENTIFIER,
snmpFddiPORTAction
        INTEGER
    }

snmpFddiPORTSMTIndex OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION

```

```

        "The value of the SMT index associated with this
        PORT."
 ::= { snmpFddiPORTEntry 1 }

snmpFddiPORTIndex OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A unique value for each PORT within a given SMT.
        Its value ranges between 1 and the sum of the
        values of snmpFddiSMTNonMasterCt
        { snmpFddiSMTEntry 6 } and snmpFddiSMTMasterCt
        { snmpFddiSMTEntry 7 } on the given SMT. The
        value for each PORT must remain constant at least
        from one re-initialization of the entity's network
        management system to the next re-initialization."
 ::= { snmpFddiPORTEntry 2 }

snmpFddiPORTPCType OBJECT-TYPE
    SYNTAX  INTEGER { a(1), b(2), s(3), m(4) }
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "PC_Type (refer to ANSI SMT 9.2.2 and ANSI SMT
        9.6.3.2)."
    REFERENCE
        "ANSI { fddiPORT 12 }"
 ::= { snmpFddiPORTEntry 3 }

snmpFddiPORTPCNeighbor OBJECT-TYPE
    SYNTAX  INTEGER { a(1), b(2), s(3), m(4), unknown(5) }
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The type (PC_Neighbor) of the remote PORT that is
        determined in PC_Signaling in R_Val (1,2) (refer
        to ANSI SMT 9.6.3.2)."
    REFERENCE
        "ANSI { fddiPORT 13 }"
 ::= { snmpFddiPORTEntry 4 }

snmpFddiPORTConnectionPolicies OBJECT-TYPE
    SYNTAX  INTEGER (0..7)
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "A value that indicates the node's PORT policies.

```

Pc-MAC-LCT, Pc-MAC-Loop, and Pc-MAC-Placement indicate how the respective PC Signaling Capability flags should be set (refer to ANSI SMT 9.4.3.2).

The value is a sum. This value initially takes the value zero, then for each PORT policy, 2 raised to a power is added to the sum. The powers are according to the following table:

Policy	Power
Pc-MAC-LCT	0
Pc-MAC-Loop	1
Pc-MAC-Placement	2

REFERENCE

"ANSI { fddiPORT 14 }"
 ::= { snmpFddiPORTEntry 5 }

snmpFddiPORTRemoteMACIndicated OBJECT-TYPE

SYNTAX INTEGER { true(1), false(2) }

ACCESS read-only

STATUS mandatory

DESCRIPTION

"The indication, in PC-Signaling that the remote partner intends to place a MAC in the output token PATH of this PORT. Signaled as R_Val (9) (refer to ANSI SMT 9.6.3.2)."

REFERENCE

"ANSI { fddiPORT 15 }"
 ::= { snmpFddiPORTEntry 6 }

snmpFddiPORTCEState OBJECT-TYPE

SYNTAX INTEGER {
 ce0(1), -- Isolated
 ce1(2), -- Insert_P
 ce2(3), -- Insert_S
 ce3(4), -- Insert_X
 ce4(5) -- Local
 }

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Indicates the current state of PORT's Configuration Element (CE) (refer to ANSI 9.7.5). Note that this value represents the Current Path information for this PORT."

REFERENCE

"ANSI { fddiPORT 16 }"

```
::= { snmpFddiPORTEntry 7 }
```

snmpFddiPORTPathsRequested OBJECT-TYPE

SYNTAX INTEGER (0..15)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"A value that indicates the desired association(s) of the port with a station PATH. The 'Primary' Path is the default. The value of 'Secondary' is only meaningful for S (slave) or M (master) PORT PC-Types. This value effects the setting of the CF_Insert_S, and CF_Insert_L flags (refer to ANSI Section 9.4.3). If the 'Primary' PATH is present, then the Primary PATH (the default PATH) is selected. If the 'Secondary' PATH is present and the 'Primary' PATH is not present, then the CF_Insert_S flag is set. If the 'Local' PATH is sent and neither the 'Primary' or 'Secondary' PATHs are sent, then the CF_Insert_L flag is set.

The value is a sum. This value initially takes the value zero, then for each type of PATH desired, 2 raised to a power is added to the sum. The powers are according to the following table:

Path	Power
Primary	0
Secondary	1
Local	2
Isolated	3 "

REFERENCE

"ANSI { fddiPORT 17 }"

```
::= { snmpFddiPORTEntry 8 }
```

snmpFddiPORTMACPlacement OBJECT-TYPE

SYNTAX FddiResourceId -- INTEGER (0..65535)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"Indicates the upstream MAC, if any, that is associated with the PORT. The value shall be zero if there is no MAC associated with the PORT. Otherwise, the value shall be equal to the value of snmpFddiMACIndex associated with the MAC."

REFERENCE

"ANSI { fddiPORT 18 }"

```
::= { snmpFddiPORTEntry 9 }
```


snmpFddiPORTAvailablePaths OBJECT-TYPE

SYNTAX INTEGER (0..7)

ACCESS read-only

STATUS mandatory

DESCRIPTION

"A value that indicates the PATH types available for M and S PORTs.

The value is a sum. This value initially takes the value zero, then for each type of PATH that this port has available, 2 raised to a power is added to the sum. The powers are according to the following table:

Path	Power
Primary	0
Secondary	1
Local	2 "

REFERENCE

"ANSI { fddiPORT 19 }"

::= { snmpFddiPORTEntry 10 }

snmpFddiPORTMACLoopTime OBJECT-TYPE

SYNTAX FddiTime

ACCESS read-write

STATUS mandatory

DESCRIPTION

"Time for the optional MAC Local Loop, T_Next(9), which is greater-than or equal-to 200 milliseconds (refer to ANSI SMT 9.4.4.2.3)."

REFERENCE

"ANSI { fddiPORT 21 }"

::= { snmpFddiPORTEntry 11 }

snmpFddiPORTTBMax OBJECT-TYPE

SYNTAX FddiTime

ACCESS read-write

STATUS mandatory

DESCRIPTION

"TB_Max (refer to ANSI SMT 9.4.4.2.1)."

REFERENCE

"ANSI { fddiPORT 32 }"

::= { snmpFddiPORTEntry 12 }

snmpFddiPORTBSFlag OBJECT-TYPE

SYNTAX INTEGER { true(1), false(2) }

ACCESS read-only

STATUS mandatory

DESCRIPTION
 "The Break State, BS_Flag (refer to ANSI SMT 9.4.3.4)."

REFERENCE
 "ANSI { fddiPORT 33 }"
 ::= { snmpFddiPORTEntry 13 }

snmpFddiPORTLCTFailCts OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "The count of the consecutive times the link confidence test (LCT) has failed during connection management (refer to ANSI 9.4.1)."

REFERENCE
 "ANSI { fddiPORT 42 }"
 ::= { snmpFddiPORTEntry 14 }

snmpFddiPORTLerEstimate OBJECT-TYPE
 SYNTAX INTEGER (4..15)
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "A long term average link error rate. It ranges from 10**-4 to 10**-15 and is reported as the absolute value of the exponent of the estimate."

REFERENCE
 "ANSI { fddiPORT 51 }"
 ::= { snmpFddiPORTEntry 15 }

snmpFddiPORTLemRejectCts OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "A link error monitoring count of the times that a link has been rejected."

REFERENCE
 "ANSI { fddiPORT 52 }"
 ::= { snmpFddiPORTEntry 16 }

snmpFddiPORTLemCts OBJECT-TYPE
 SYNTAX Counter
 ACCESS read-only
 STATUS mandatory
 DESCRIPTION
 "The aggregate link error monitor error count, set

to zero only on station power_up."

REFERENCE

"ANSI { fddiPORT 53 }"

::= { snmpFddiPORTEntry 17 }

snmpFddiPORTLerCutoff OBJECT-TYPE

SYNTAX INTEGER (4..15)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The link error rate estimate at which a link connection will be broken. It ranges from 10^{*-4} to 10^{*-15} and is reported as the absolute value of the exponent."

REFERENCE

"ANSI { fddiPORT 58 }"

::= { snmpFddiPORTEntry 18 }

snmpFddiPORTLerAlarm OBJECT-TYPE

SYNTAX INTEGER (4..15)

ACCESS read-write

STATUS mandatory

DESCRIPTION

"The link error rate estimate at which a link connection will generate an alarm. It ranges from 10^{*-4} to 10^{*-15} and is reported as the absolute value of the exponent of the estimate."

REFERENCE

"ANSI { fddiPORT 59 }"

::= { snmpFddiPORTEntry 19 }

snmpFddiPORTConnectState OBJECT-TYPE

SYNTAX INTEGER {

disabled(1),

connecting(2),

standby(3),

active(4)

}

ACCESS read-only

STATUS mandatory

DESCRIPTION

"An indication of the connect state of this PORT. Basically, this gives a higher level view of the state of the connection by grouping PCM states and the PC-Withhold flag state. The supported values and their corresponding PCM states and PC-Withhold condition, when relevant, are:

disabled: (PC0:Off, PC9:Maint)

connecting: (PC1(Break) || PC3 (Connect) || PC4
(Next) || PC5 (Signal) || PC6
(Join) || PC7 (Verify)) &&
(PC_Withhold = None)

standby: (NOT PC_Withhold == None)

active: (PC2:Trace || PC8:Active) "

REFERENCE

"ANSI { fddiPORT 61 }"
::= { snmpFddiPORTEntry 20 }

snmpFddiPORTPCMState OBJECT-TYPE

SYNTAX INTEGER {
 pc0(1), -- Off
 pc1(2), -- Break
 pc2(3), -- Trace
 pc3(4), -- Connect
 pc4(5), -- Next
 pc5(6), -- Signal
 pc6(7), -- Join
 pc7(8), -- Verify
 pc8(9), -- Active
 pc9(10) -- Maint
}

ACCESS read-only

STATUS mandatory

DESCRIPTION

"(refer to SMT 9.6.2)."

REFERENCE

"ANSI { fddiPORT 62 }"
::= { snmpFddiPORTEntry 21 }

snmpFddiPORTPCWithhold OBJECT-TYPE

SYNTAX INTEGER { none(1), m-m(2), other(3) }

ACCESS read-only

STATUS mandatory

DESCRIPTION

"PC_Withhold, (refer to ANSI SMT 9.4.1)."

REFERENCE

"ANSI { fddiPORT 63 }"
::= { snmpFddiPORTEntry 22 }

snmpFddiPORTLerCondition OBJECT-TYPE

SYNTAX INTEGER { true(1), false(2) }

ACCESS read-only

STATUS mandatory

DESCRIPTION

"This variable is set to true whenever LerEstimate is less than or equal to LerAlarm."

REFERENCE

"ANSI { fddiPORT 64 }"
::= { snmpFddiPORTEntry 23 }

snmpFddiPORTChipSet OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

ACCESS read-only

STATUS mandatory

DESCRIPTION

"This object identifies the hardware chip(s) which is (are) principally responsible for the implementation of the PORT (PHY) function. A few OBJECT IDENTIFIERS are identified elsewhere in this memo. For those The assignment of additional OBJECT IDENTIFIERS to various types of hardware chip sets is managed by the IANA. For example, vendors whose chip sets are not defined in this memo may request a number from the Internet Assigned Numbers Authority (IANA) which indicates the assignment of a enterprise specific subtree which, among other things, may be used to allocate OBJECT IDENTIFIER assignments for that enterprise's chip sets. Similarly, in the absence of an appropriately assigned OBJECT IDENTIFIER in this memo or in an enterprise specific subtree of a chip vendor, a board or system vendor can request a number for a subtree from the IANA and make an appropriate assignment. It is desired that, whenever possible, the same OBJECT IDENTIFIER be used for all chips of a given type. Consequently, the assignment made in this memo for a chip, if any, should be used in preference to any other assignment and the assignment made by the chip manufacturer, if any, should be used in preference to assignments made by users of those chips. If the hardware chip set is unknown, the object identifier

unknownChipSet OBJECT IDENTIFIER ::= { 0 0 }

is returned. Note that unknownChipSet is a syntactically valid object identifier, and any conformant implementation of ASN.1 and the BER must be able to generate and recognize this

```

        value."
 ::= { snmpFddiPORTEntry 24 }

snmpFddiPORTAction OBJECT-TYPE
    SYNTAX  INTEGER {
        other(1),           -- none of the following
        maintPORT(2),
        enablePORT(3),
        disablePORT(4),
        startPORT(5),
        stopPORT(6)
    }
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "This object, when read, always returns a value of
        other(1).  The behavior of setting this variable
        to each of the acceptable values is as follows:

        Other:           Results in a badValue error.

        maintPORT:       Signal PC_Maint

        enablePORT:      Signal PC_Enable

        disablePORT:     Signal PC_Disable

        startPORT:       Signal PC_Start

        stopPORT:        Signal PC_Stop

        Signals cause an SM_CM_CONTROL.request service to
        be generated with a control_action of 'Signal' and
        the 'variable' parameter set with the appropriate
        value (i.e., PC_Maint, PC_Enable, PC_Disable,
        PC_Start, PC_Stop).  Ref. ANSI SMT Section 9.3.2.

        Attempts to set this object to all other values
        results in a badValue error."
    REFERENCE
        "ANSI { fddiPORT 70 }"
 ::= { snmpFddiPORTEntry 25 }

-- the ATTACHMENT group
-- Implementation of the ATTACHMENT group is mandatory for
-- all systems which implement manageable FDDI subsystems.

```

```

snmpFddiATTACHMENTNumber OBJECT-TYPE
    SYNTAX  INTEGER (0..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The total number of attachments (across all SMTs)
        on this network management application entity.
        The value for this variable must remain constant
        at least from one re-initialization of the
        entity's network management system to the next
        re-initialization."
    ::= { snmpFddiATTACHMENT 1 }

-- the ATTACHMENT table

snmpFddiATTACHMENTTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF SnmpFddiATTACHMENTEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A list of ATTACHMENT entries. The number of
        entries is given by the value of
        snmpFddiATTACHMENTNumber."
    ::= { snmpFddiATTACHMENT 2 }

snmpFddiATTACHMENTEntry OBJECT-TYPE
    SYNTAX  SnmpFddiATTACHMENTEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "An ATTACHMENT entry containing information common
        to a given set of ATTACHMENTS.

        The ATTACHMENT Resource represents a PORT or a
        pair of PORTs plus the optional associated optical
        bypass that are managed as a functional unit.
        Because of its relationship to the PORT Objects,
        there is a natural association of ATTACHMENT
        Resource Indices to the PORT Indices. The
        resource index for the ATTACHMENT is equal to the
        associated PORT index for 'single-attachment' and
        'concentrator' type snmpFddiATTACHMENTClasses.
        For 'dual-attachment' Classes, the ATTACHMENT
        Index is the PORT Index of the A PORT of the A/B
        PORT Pair that represents the ATTACHMENT."
    INDEX   { snmpFddiATTACHMENTSMTIndex,
               snmpFddiATTACHMENTIndex }

```

```

 ::= { snmpFddiATTACHMENTTable 1 }

SnmpFddiATTACHMENTEntry ::=
    SEQUENCE {
        snmpFddiATTACHMENTSMTIndex
            INTEGER,
        snmpFddiATTACHMENTIndex
            INTEGER,
        snmpFddiATTACHMENTClass
            INTEGER,
        snmpFddiATTACHMENTOpticalBypassPresent
            INTEGER,
        snmpFddiATTACHMENTIMaxExpiration
            FddiTime,
        snmpFddiATTACHMENTInsertedStatus
            INTEGER,
        snmpFddiATTACHMENTInsertPolicy
            INTEGER
    }

snmpFddiATTACHMENTSMTIndex OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The value of the SMT index associated with this
        ATTACHMENT."
    ::= { snmpFddiATTACHMENTEntry 1 }

snmpFddiATTACHMENTIndex OBJECT-TYPE
    SYNTAX  INTEGER (1..65535)
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A unique value for each ATTACHMENT on a given
        SMT.  Its value ranges between 1 and the sum of
        the values of snmpFddiSMTNonMasterCt {
        snmpFddiSMTEntry 6 } and snmpFddiSMTMasterCt {
        snmpFddiSMTEntry 7 } on the given SMT.  The value
        for each ATTACHMENT must remain constant at least
        from one re-initialization of the entity's network
        management system to the next re-initialization."
    ::= { snmpFddiATTACHMENTEntry 2 }

snmpFddiATTACHMENTClass OBJECT-TYPE
    SYNTAX  INTEGER {
        single-attachment(1),
        dual-attachment(2),

```



```

        concentrator(3)
    }
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "The Attachment class. This represents a PORT or
    a pair of PORTs plus the associated optional
    optical bypass that are managed as a functional
    unit. The PORT associations are the following:

        single-attachment - S PORTs
        dual-attachment - A/B PORT Pairs
        concentrator - M PORTs "
```

REFERENCE

```

    "ANSI { fddiATTACHMENT 11 }"
    ::= { snmpFddiATTACHMENTEntry 3 }
```

snmpFddiATTACHMENTOpticalBypassPresent OBJECT-TYPE

```

SYNTAX INTEGER { true(1), false(2) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "The value of this value is false for 'single-
    attachment' and { snmpFddiATTACHMENT 11 }.
    Correct operation of CMT for single-attachment and
    concentrator attachments requires that a bypass
    function must not loopback the network side of the
    MIC, but only the node side."
```

REFERENCE

```

    "ANSI { fddiATTACHMENT 12 }"
    ::= { snmpFddiATTACHMENTEntry 4 }
```

snmpFddiATTACHMENTIMaxExpiration OBJECT-TYPE

```

SYNTAX FddiTime
ACCESS read-only
STATUS mandatory
DESCRIPTION
    "I_Max (refer to ANSI SMT 9.4.4.2.1). It is
    recognized that some currently deployed systems do
    not implement an optical bypass. Systems which do
    not implement optical bypass should return a value
    of 0."
```

REFERENCE

```

    "ANSI { fddiATTACHMENT 13 }"
    ::= { snmpFddiATTACHMENTEntry 5 }
```

snmpFddiATTACHMENTInsertedStatus OBJECT-TYPE

```

SYNTAX INTEGER { true(1), false(2), unimplemented(3) }
```

```

ACCESS    read-only
STATUS    mandatory
DESCRIPTION
    "Indicates whether the attachment is currently
    inserted in the node."
REFERENCE
    "ANSI { fddiATTACHMENT 14 }"
    ::= { snmpFddiATTACHMENTEntry 6 }

snmpFddiATTACHMENTInsertPolicy OBJECT-TYPE
SYNTAX    INTEGER { true(1), false(2), unimplemented(3) }
ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "Indicates the Insert Policy for this Attachment.
    Insert: True (1), Don't Insert: False (2),
    Unimplemented (3)"
REFERENCE
    "ANSI { fddiATTACHMENT 15 }"
    ::= { snmpFddiATTACHMENTEntry 7 }

-- the Chip Set group
-- The following object identifiers are allocated for use
-- with the snmpFddiMACChipSet and snmpFddiPORTChipSet
-- variables.

    snmpFddiPHYChipSets    -- Chips primarily responsible
                           -- for implementing the PHY
                           -- function.
                           OBJECT IDENTIFIER ::= { snmpFddiChipSets 1 }

                           -- None defined at present
                           -- Chipsets may someday be
                           -- defined here

    snmpFddiMACChipSets    -- Chips primarily responsible
                           -- for implementing the
                           -- MAC function.
                           OBJECT IDENTIFIER ::= { snmpFddiChipSets 2 }

                           -- None defined at present
                           -- Chipsets may someday be
                           -- defined here

    snmpFddiPHYMACChipSets -- Chips which implement both
                           -- the PHY and MAC functions

```

OBJECT IDENTIFIER ::= { snmpFddiChipSets 3 }

-- None defined at present
-- Chipsets may someday be
-- defined here

END

6. Acknowledgements

This document was produced by the IETF FDDI MIB working group:

Steve Adams, Digital Equipment Corporation
Hossein Alaee, 3Com Corporation
Haggar Alsaleh, Bell Northern Research
William Anderson, Mitre Corporation
Alan Apt, Addison-Wesley
Mary Artibee, Silicon Graphics
Karen Auerbach, Epilogue Technologies
Doug Bagnall, Apollo/Hewlett Packard
Chet Birger, Coral Network Corporation
Pablo Brenner, Sparta
Howard Brown, Cabletron
Jack Brown, US Army Computer Engineering Center
Eric Brunner
Jeff Case, The University of Tennessee
Tammy Chan, Fibercom
Asheem Chandna, AT&T
Cho Y. Chang, Apollo/Hewlett Packard
Chris Chiotasso, Fibronics
Paul Ciarfella, Digital Equipment Corporation
John Cook, Chipcom
Don Coolidge, Silicon Graphics
Burt Cyr, Unisys
James R. Davin, Massachusetts Institute of Technology
Nabil Damouny
Nadya El-Afandi, Network Systems Corporation
Hunaid Engineer, Cray Research
Jeff Fitzgerald, Fibercom
Richard Fox, Synoptics
Stan Froyd, ACC
Debbie Futch, U.S. Naval Surface Warfare Center
Joseph Golio, Cray Research
Jeremy Greene, Coral
Brian D. Handspicker, Digital Equipment Corporation
Peter Hayden, Digital Equipment Corporation
Scott Hiles, U.S. Naval Surface Warfare Center
Greg Jones, Data General

Satish Joshi, SynOptics Communications
Jayant Kadambi, AT&T Bell Labs
Joanna Karwowska, Data General
Frank Kastenholz, Interlan
Jim Kinder, Fibercom
Christopher Kolb, PSI
Cheryl Krupczak, NCR
Peter Lin, Vitalink
Then Liu
John R. LoVerso, Concurrent Computer Corporation
Ron Mackey
Gary Malkin, Proteon
Bruce McClure, Synernetics
Keith McCloghrie, Hughes Lan Systems
Donna McMaster, SynOptics
John O'Hara, Massachusetts Institute of Technology
Dave Perkins, SynOptics Communications
James E. Reeves, SynOptics Communications
Jim Reinstedler, Ungermann-Bass
Radhi Renous, Fibronics
Anil Rijsinghani, Digital Equipment Corporation
Bob Rolla, Synernetics
Nelson Ronkin, Synernetics
Marshall T. Rose, Performance Systems International, Inc.
Milt Roselinsky, CMC
Jon Saperia, Digital Equipment Corporation
Greg Satz, cisco Systems
Steven Senum, Network Systems Corporation
Jim Sheridan, IBM Corporation
Jeffrey Schiller, MIT
Dror Shindelman, Sparta
Mark Sleeper, Sparta
Craig Smelser, Digital Equipment Corporation
Lou Steinberg, IBM Corporation
Mary Jane Strohl, Apollo/Hewlett Packard
Sally Tarquinio, Mitre Corporation
Kaj Tesink, Bellcore
Ian Thomas, Chipcom
Dean Throop, Data General
Bill Townsend, Xylogics
Ahmet H. Tuncay, SynOptics Communications
Mike Turico, Motorola
Chris VandenBerg, ACC
Sudhanshu Verma, Hewlett Packard
Joe Vermeulen, UNISYS
David Waiteman, BBN
Bert Williams, Synernetics
Mark Wood, AT&T Computer Systems

Y. C. Yang
Denis Yaro, Sun Microsystems
Jeff Young, Cray Research

The editor gratefully acknowledges the contributions of the editor of the ANSI X3T9.5 SMT document, Mary Jane Strohl of Hewlett Packard/Apollo, whose provision of that document in machine readable form saved much typing and avoided many data entry errors.

The author gratefully acknowledges the labors of Dr. Marshall T. Rose in assisting with converting this document to the new concise MIB format.

8. References

- [1] Rose M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based internets", RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.
- [2] McCloghrie K., and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets", RFC 1156, Hughes LAN Systems, Performance Systems International, May 1990.
- [3] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.
- [4] McCloghrie K., and M. Rose, Editors, "Management Information Base for Network Management of TCP/IP-based internets", RFC 1213, Performance Systems International, March 1991.
- [5] Information processing systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), International Organization for Standardization, International Standard 8824, December 1987.
- [6] Information processing systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Notation One (ASN.1), International Organization for Standardization, International Standard 8825, December 1987.
- [7] Rose, M., and K. McCloghrie, Editors, "Concise MIB Definitions", RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.
- [8] American National Standards Institute, "FDDI Station Management (SMT)", Preliminary Draft Proposed American National Standard,

American National Standards Institute, X3T9/90-X3T9.5/84-49 REV
6.2, May 18, 1990.

Security Considerations

Security issues are not discussed in this memo.

Author's Address

Jeffrey D. Case
SNMP Research, Incorporated
3001 Kimberlin Heights Road
Knoxville, Tennessee 37920

Phone: (615) 573-1434

EMail: case@CS.UTK.EDU