Chapter 2

MIB Definition

RFC1213-MIB { iso org(3) dod(6) internet(1) mgmt(2) 1 }

DEFINITIONS ::= BEGIN

```
IMPORTS
     mgmt, NetworkAddress, IpAddress, Counter, Gauge,TimeTicks
          FROM RFC1155-SMI
     OBJECT-TYPE
          FROM RFC-1212;
```

This MIB module uses the extended OBJECT-TYPE macro as defined in [14];

MIB-II (same prefix as MIB-I)

```
mib-2                    OBJECT IDENTIFIER ::=    { mgmt 1 }
```

textual conventions

DisplayString ::=
     OCTET STRING
     This data type is used to model textual information taken from the NVT ASCII character set. By convention, objects with this syntax are declared as having

     SIZE (0..255)

PhysAddress ::=
     OCTET STRING
     This data type is used to model media addresses. For many types of media, this will be in a binary representation. For example, an ethernet address would be represented as a string of 6

octets.

```
groups in MIB-II

system      OBJECT IDENTIFIER ::=      { mib-2 1 }

interfaces  OBJECT IDENTIFIER ::=      { mib-2 2 }

at                      OBJECT IDENTIFIER ::=    { mib-2 3 }

ip                      OBJECT IDENTIFIER ::=    { mib-2 4 }

icmp                    OBJECT IDENTIFIER ::=    { mib-2 5 }

tcp         OBJECT IDENTIFIER ::=      { mib-2 6 }

udp                     OBJECT IDENTIFIER ::=    { mib-2 7 }

egp         OBJECT IDENTIFIER ::=      { mib-2 8 }

—historical (some say hysterical)
—cmot                   OBJECT IDENTIFIER ::=    { mib-2 9 }

transmission            OBJECT IDENTIFIER ::=    { mib-2 10 }

snmp                    OBJECT IDENTIFIER ::=    { mib-2 11 }

private     OBJECT IDENTIFIER ::=      { internet 4 }
enterprisesOBJECT IDENTIFIER ::=       { private 1 }

system      OBJECT IDENTIFIER ::=      { mib 1 }
interfaces  OBJECT IDENTIFIER ::=      { mib 2 }
at                      OBJECT IDENTIFIER ::=    { mib 3 }
ip                      OBJECT IDENTIFIER ::=    { mib 4 }
icmp                    OBJECT IDENTIFIER ::=    { mib 5 }
tcp         OBJECT IDENTIFIER ::=      { mib 6 }
udp                     OBJECT IDENTIFIER ::=    { mib 7 }
egp         OBJECT IDENTIFIER ::=      { mib 8 }

synoptics               OBJECT IDENTIFIER ::=    { enterprises 45 }
products                OBJECT IDENTIFIER ::=    {synoptics 1 }
registration            OBJECT IDENTIFIER ::=    {synoptics 3 }

temporary   OBJECT IDENTIFIER ::=      {synoptics 2 }
```

```
series1000 OBJECT IDENTIFIER ::=          {products 1}
s3snmpAgent              OBJECT IDENTIFIER ::=     {products 2}
series3000 OBJECT IDENTIFIER ::=          {products 3}

ieee8023                 OBJECT IDENTIFIER ::=     {temporary 1}
```

```
s3000Chassis                OBJECT IDENTIFIER ::=    {series3000 1}
s3000Ethernet               OBJECT IDENTIFIER ::=    {series3000 2}
s3000TokenRing              OBJECT IDENTIFIER ::=    {series3000 3}

s3000Bridge                 OBJECT IDENTIFIER ::=    {series3000 5}

s3agentSW                   OBJECT IDENTIFIER ::=    {s3snmpAgent 1}
s3agentNetProtocol          OBJECT IDENTIFIER ::=    {s3snmpAgent 2}
trapRcvrTable               OBJECT IDENTIFIER ::=    {s3snmpAgent 3}
s3agentHW                   OBJECT IDENTIFIER ::=    {s3snmpAgent 4}
s3agentSpecific             OBJECT IDENTIFIER ::=    {s3snmpAgent 5}

s3agentIpProtocol           OBJECT IDENTIFIER ::=    {s3agentNetProtocol 1}

s3recother  OBJECT IDENTIFIER ::=        {registration 1}
s3rec3000   OBJECT IDENTIFIER ::=        {registration 2}
s3rec3030   OBJECT IDENTIFIER ::=        {registration 3}
s3rec3138   OBJECT IDENTIFIER ::=        {registration 4}

s3rec332X   OBJECT IDENTIFIER ::=        {registration 5}
s3rec332XS                  OBJECT IDENTIFIER ::=    {registration 6}
s3rec3356   OBJECT IDENTIFIER ::=        {registration 7}

—Common Bridge Objects
brdgcommon                  OBJECT IDENTIFIER ::=    {s3000Bridge 1 }

—Remote Bridge Objects
oemid       OBJECT IDENTIFIER ::=        {s3000Bridge 3 }

retix                       OBJECT IDENTIFIER ::=    {enterprises 72 }
product                     OBJECT IDENTIFIER ::=    {retix 8 }
prembridge                  OBJECT IDENTIFIER ::=    {product 99 }

station     OBJECT IDENTIFIER ::=        { oemid 1 } — Station Group
lapb                        OBJECT IDENTIFIER ::=    { oemid 2 } — LAPB Group
ieee8023                    OBJECT IDENTIFIER ::=    { oemid 3 } — IEEE 802.3 Group
phySerIf                    OBJECT IDENTIFIER ::=    { oemid 4 } — Physical Serial
Group
mlink                       OBJECT IDENTIFIER ::=    { oemid 5 } — Multilink Group
lan         OBJECT IDENTIFIER ::=        { oemid 6 } — LAN Group
bridge      OBJECT IDENTIFIER ::=        { oemid 7 } — Bridge Group
product                     OBJECT IDENTIFIER ::=    { oemid 8 } — Product Group
```

```
br12                            OBJECT IDENTIFIER ::=    { oemid 9 } — Br12 Bridge Group
router      OBJECT IDENTIFIER ::=        { oemid 10 } — Router Group
boot                            OBJECT IDENTIFIER ::=    { oemid 11 } — Bootp Group
boothelper OBJECT IDENTIFIER ::=         { oemid 12 } — Boothelper Group
remote      OBJECT IDENTIFIER ::=        { oemid 13 } — Remote Group
ipx         OBJECT IDENTIFIER ::=        { remote 1 } — IPX group
decnet      OBJECT IDENTIFIER ::=        { remote 2 } — Decnet group
rmtLapb                         OBJECT IDENTIFIER ::=    { remote 3 } — Remote Lapb group
x25                             OBJECT IDENTIFIER ::=    { remote 4 } — x25 group
```

```
— object types

    — the Product group

    p4942  OBJECT IDENTIFIER ::=        { product 1 }
    p2244m2                     OBJECT IDENTIFIER ::=    { product 2 }
    p2265  OBJECT IDENTIFIER ::=        { product 3 }
    p4660  OBJECT IDENTIFIER ::=        { product 4 }
    p4820  OBJECT IDENTIFIER ::=        { product 5 }
    p4880  OBJECT IDENTIFIER ::=        { product 6 }
    p4850  OBJECT IDENTIFIER ::=        { product 7 }
    p4760  OBJECT IDENTIFIER ::=        { product 8 }
    p4941  OBJECT IDENTIFIER ::=        { product 9 }
```

System grouptc "System group"§

Implementation of the System group is mandatory for all systems. If an agent is not configured to have a value for any of these variables, a string of length 0 is returned.

```
sysDescr OBJECT-TYPE
    SYNTAX              DisplayString (SIZE (0..255))
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “A textual description of the entity.  This value should include the full name and version
        identification of the system’s hardware type, software operating-system, and networking
        software.  It is mandatory that this only contain printable ASCII characters.”
    ::= { system 1 }

sysObjectID OBJECT-TYPE
    SYNTAX              OBJECT IDENTIFIER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The vendor’s authoritative identification of the network management subsystem
        contained in the entity.  This value is allocated within the SMI enterprises subtree
        (1.3.6.1.4.1) and provides an easy and unambiguous means for determining ‘what kind of
        box’ is being managed.  For example, if vendor ‘Flintstones, Inc.’ was assigned the
        subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its ‘Fred
```

```
          Router'."
     ::= { system 2 }
```

```
sysUpTime OBJECT-TYPE
    SYNTAX              TimeTicks
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The time (in hundredths of a second) since the network management portion of the
        system was last re-initialized.”
    ::= { system 3 }

sysContact OBJECT-TYPE
    SYNTAX              DisplayString (SIZE (0..255))
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “The textual identification of the contact person for this managed node, together with
        information on how to contact this person.”
    ::= { system 4 }

sysName OBJECT-TYPE
    SYNTAX              DisplayString (SIZE (0..255))
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “An administratively-assigned name for this managed node.  By convention, this is the
        node’s fully-qualified domain name.”
    ::= { system 5 }

sysLocation OBJECT-TYPE
    SYNTAX              DisplayString (SIZE (0..255))
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “The physical location of this node (e.g., `telephone closet, 3rd floor’).”
    ::= { system 6 }

sysServices OBJECT-TYPE
    SYNTAX              INTEGER (0..127)
    ACCESS              read-only
    STATUS              mandatory
```

DESCRIPTION
"A value which indicates the set of services that this entity primarily offers. The value is a sum. This sum initially takes the value zero, Then, for each layer, L, in the range 1 through 7, that this node performs transactions for, 2 raised to (L - 1) is added to the sum. For example, a node which performs primarily routing functions would have a value of 4 (2^(3-1)). In contrast, a node which is a host offering application services would have a value of 72 (2^(4-1) + 2^(7-1)). Note that in the context of the Internet suite of protocols, values should be calculated accordingly:

layer	functionality
1	physical (e.g., repeaters)
2	datalink/subnetwork (e.g., bridges)
3	internet (e.g., IP gateways)
4	end-to-end (e.g., IP hosts)
7	applications (e.g., mail relays)

For systems including OSI protocols, layers 5 and 6 may also be counted."
::= { system 7 }

Interfaces grouptc "Interfaces group"§

Implementation of the Interfaces group is mandatory for all systems.

```
ifNumber OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory
```

DESCRIPTION
"The number of network interfaces (regardless of their current state) present on this system."
::= { interfaces 1 }

the Interfaces tabletc "the Interfaces table"§

The Interfaces table contains information on the entity's interfaces. Each interface is thought of as being attached to a 'subnetwork'. Note that this term should not be confused with 'subnet' which refers to an addressing partitioning scheme used in the Internet suite of protocols.

```
ifTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF IfEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “A list of interface entries. The number of entries is given by the value of ifNumber.”
    ::= { interfaces 2 }

ifEntry OBJECT-TYPE
    SYNTAX              IfEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “An interface entry containing objects at the subnetwork layer and below for a particular
        interface.”
    INDEX                   { ifIndex }
    ::= { ifTable 1 }
```

```
IfEntry ::=
    SEQUENCE {
        ifIndex
            INTEGER,
        ifDescr
            DisplayString,
        ifType
            INTEGER,
        ifMtu
            INTEGER,
        ifSpeed
            Gauge,
        ifPhysAddress
            PhysAddress,
        ifAdminStatus
            INTEGER,
        ifOperStatus
            INTEGER,
        ifLastChange
            TimeTicks,
        ifInOctets
            Counter,
        ifInUcastPkts
            Counter,
        ifInNUcastPkts
            Counter,
        ifInDiscards
            Counter,
        ifInErrors
            Counter,
        ifInUnknownProtos
            Counter,
        ifOutOctets
            Counter,
        ifOutUcastPkts
            Counter,
        ifOutNUcastPkts
            Counter,
        ifOutDiscards
            Counter,
        ifOutErrors
```

```
            Counter,
        ifOutQLen
            Gauge,
        ifSpecific
            OBJECT IDENTIFIER
    }

ifIndex OBJECT-TYPE
    SYNTAX                      INTEGER
    ACCESS                      read-only
    STATUS                      mandatory

    DESCRIPTION
        “A unique value for each interface. Its value ranges between 1 and the value of
        ifNumber. The value for each interface must remain constant at least from one re-
        initialization of the entity’s network management system to the next re- initialization.”
    ::= { ifEntry 1 }

ifDescr OBJECT-TYPE
    SYNTAX                      DisplayString (SIZE (0..255))
    ACCESS                      read-only
    STATUS                      mandatory

    DESCRIPTION
        “A textual string containing information about the interface.  This string should include
        the name of the manufacturer, the product name and the version of the hardware
        interface.”
    ::= { ifEntry 2 }

ifType OBJECT-TYPE
   SYNTAX                       INTEGER {
       other(1),                none of the following
       regular1822(2),
       hdh1822(3),
       ddn-x25(4),
       rfc877-x25(5),
       ethernet-csmacd(6),
       iso88023-csmacd(7),
       iso88024-tokenBus(8),
       iso88025-tokenRing(9),
       iso88026-man(10),
       starLan(11),
       proteon-10Mbit(12),
            proteon-80Mbit(13),
            hyperchannel(14),
            fddi(15),
            lapb(16),
            sdlc(17),
            ds1(18),                 T-1
            e1(19),                  european equiv. of T-1
            basicISDN(20),
            primaryISDN(21),         proprietary serial
            propPointToPointSerial(22),
            ppp(23),
            softwareLoopback(24),
            eon(25),                 CLNP over IP [11]
            ethernet-3Mbit(26),
            nsip(27),                XNS over IP
            slip(28),                generic SLIP
            ultra(29),               ULTRA technologies
            ds3(30),                 T-3
            sip(31),                 SMDS
            frame-relay(32)
            }
     ACCESS                          read-only
      STATUS                         mandatory

      DESCRIPTION
           "The type of interface, distinguished according to the physical/link protocol(s)
           immediately `below' the network layer in the protocol stack."
      ::= { ifEntry 3 }

ifMtu OBJECT-TYPE
      SYNTAX                         INTEGER
      ACCESS                         read-only
      STATUS                         mandatory

      DESCRIPTION
           "The size of the largest datagram which can be sent/received on the interface, specified in
           octets.  For interfaces that are used for transmitting network datagrams, this is the size of
           the largest network datagram that can be sent on the interface."
      ::= { ifEntry 4 }

ifSpeed OBJECT-TYPE
      SYNTAX                         Gauge
```

```
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “An estimate of the interface’s current bandwidth in bits per second.  For interfaces
        which do not vary in bandwidth or for those where no accurate estimation can be made,
        this object should contain the nominal bandwidth.”
    ::= { ifEntry 5 }

ifPhysAddress OBJECT-TYPE
    SYNTAX              PhysAddress
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The interface’s address at the protocol layer immediately `below’ the network layer in
        the protocol stack.  For interfaces which do not have such an address (e.g., a serial line),
        this object should contain an octet string of zero length.”
    ::= { ifEntry 6 }

ifAdminStatus OBJECT-TYPE
    SYNTAX              INTEGER {
        up(1),          ready to pass packets
        down(2),
        testing(3)      in some test mode
        }
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The desired state of the interface.  The testing(3) state indicates that no operational
        packets can be passed.”
    ::= { ifEntry 7 }

ifOperStatus OBJECT-TYPE
    SYNTAX              INTEGER {
        up(1),          ready to pass packets
        down(2),
        testing(3)      in some test mode
        }
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        “The current operational state of the interface. The testing(3) state indicates that no
        operational packets can be passed.”
    ::= { ifEntry 8 }

ifLastChange OBJECT-TYPE
    SYNTAX              TimeTicks
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The value of sysUpTime at the time the interface entered its current operational state.  If
        the current state was entered prior to the last re- initialization of the local network
        management subsystem, then this object contains a zero value.”
    ::= { ifEntry 9 }

ifInOctets OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of octets received on the interface, including framing characters.”
    ::= { ifEntry 10 }

ifInUcastPkts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of subnetwork-unicast packets delivered to a higher-layer protocol.”
    ::= { ifEntry 11 }

ifInNUcastPkts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of non-unicast (i.e., subnetwork- broadcast or subnetwork-multicast)
        packets delivered to a higher-layer protocol.”
    ::= { ifEntry 12 }

ifInDiscards OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of inbound packets which were chosen to be discarded even though no
        errors had been detected to prevent their being deliverable to a higher-layer protocol.
        One possible reason for discarding such a packet could be to free up buffer space.”
    ::= { ifEntry 13 }

ifInErrors OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of inbound packets that contained errors preventing them from being
        deliverable to a higher-layer protocol.”
    ::= { ifEntry 14 }

ifInUnknownProtos OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of packets received via the interface which were discarded because of an
        unknown or unsupported protocol.”
    ::= { ifEntry 15 }

ifOutOctets OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of octets transmitted out of the interface, including framing
        characters.”
    ::= { ifEntry 16 }
```

```
ifOutUcastPkts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of packets that higher-level protocols requested be transmitted to a
        subnetwork-unicast address, including those that were discarded or not sent.”
    ::= { ifEntry 17 }

ifOutNUcastPkts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of packets that higher-level protocols requested be transmitted to a
        non- unicast (i.e., a subnetwork-broadcast or subnetwork-multicast) address, including
        those that were discarded or not sent.”
    ::= { ifEntry 18 }

ifOutDiscards OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of outbound packets which were chosen to be discarded even though no
        errors had been detected to prevent their being transmitted.  One possible reason for
        discarding such a packet could be to free up buffer space.”
    ::= { ifEntry 19 }

ifOutErrors OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of outbound packets that could not be transmitted because of errors.”
    ::= { ifEntry 20 }

ifOutQLen OBJECT-TYPE
    SYNTAX              Gauge
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The length of the output packet queue (in packets)."
    ::= { ifEntry 21 }

ifSpecific OBJECT-TYPE
    SYNTAX              OBJECT IDENTIFIER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "A reference to MIB definitions specific to the particular media being used to realize the
        interface.  For example, if the interface is realized by an ethernet, then the value of this
        object refers to a document defining objects specific to ethernet.  If this information is
        not present, its value should be set to the OBJECT IDENTIFIER { 0 0 }, which is a
        syntatically valid object identifier, and any conformant implementation of ASN.1 and
        BER must be able to generate and recognize this value."
    ::= { ifEntry 22 }
```

Address Translation grouptc "Address Translation group"§

Implementation of the Address Translation group is mandatory for all systems. Note however that this group is deprecated by MIB-II. That is, it is being included solely for compatibility with MIB-I nodes, and will most likely be excluded from MIB-III nodes. From MIB-II and onwards, each network protocol group contains its own address translation tables.

The Address Translation group contains one table which is the union across all interfaces of the translation tables for converting a NetworkAddress (e.g., an IP address) into a subnetwork-specific address. For lack of a better term, this document refers to such a subnetwork-specific address as a ‘physical’ address.

Examples of such translation tables are: for broadcast media where ARP is in use, the translation table is equivalent to the ARP cache; or, on an X.25 network where non-algorithmic translation to X.121 addresses is required, the translation table contains the NetworkAddress to X.121 address equivalences.

```
atTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF AtEntry
    ACCESS              not-accessible
    STATUS              deprecated

    DESCRIPTION
        “The Address Translation tables contain the NetworkAddress to `physical’ address
        equivalences. Some interfaces do not use translation tables for determining address
        equivalences (e.g., DDN-X.25 has an algorithmic method); if all interfaces are of this
        type, then the Address Translation table is empty, i.e., has zero entries.”
    ::= { at 1 }

atEntry OBJECT-TYPE
    SYNTAX              AtEntry
    ACCESS              not-accessible
    STATUS              deprecated

    DESCRIPTION
        “Each entry contains one NetworkAddress to `physical’ address equivalence.”
    INDEX                   { atIfIndex,
    ::= { atTable 1 }
```

```
AtEntry ::=
    SEQUENCE {
        atIfIndex
            INTEGER,
        atPhysAddress
            PhysAddress,
        atNetAddress
            NetworkAddress
    }

atIfIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              deprecated

    DESCRIPTION
        “The interface on which this entry’s equivalence is effective.  The interface identified by
        a particular value of this index is the same interface as identified by the same value of
        ifIndex.”
    ::= { atEntry 1 }

atPhysAddress OBJECT-TYPE
    SYNTAX              PhysAddress
    ACCESS              read-write
    STATUS              deprecated

    DESCRIPTION
        “The media-dependent `physical’ address. Setting this object to a null string (one of zero
        length) has the effect of invaliding the corresponding entry in the atTable object.  That is,
        it effectively dissasociates the interface identified with said entry from the mapping
        identified with said entry.  It is an implementation-specific matter as to whether the agent
        removes an invalidated entry from the table. Accordingly, management stations must be
        prepared to receive tabular information from agents that corresponds to entries not
        currently in use. Proper interpretation of such entries requires examination of the relevant
        atPhysAddress object.”
    ::= { atEntry 2 }

atNetAddress OBJECT-TYPE
    SYNTAX              NetworkAddress
    ACCESS              read-write
    STATUS              deprecated
```

```
DESCRIPTION
     “The NetworkAddress (e.g., the IP address) corresponding to the media-dependent
     `physical’ address.”
::= { atEntry 3 }
```

IP grouptc "

IP group"§

Implementation of the IP group is mandatory for all systems.

```
ipForwarding OBJECT-TYPE
    SYNTAX              INTEGER {
        forwarding(1),          acting as a gateway
        not-forwarding(2)       NOT acting as a gateway }

    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The indication of whether this entity is acting as an IP gateway in respect to the
        forwarding of datagrams received by, but not addressed to, this entity.  IP gateways
        forward datagrams.  IP hosts do not (except those source-routed via the host). Note that
        for some managed nodes, this object may take on only a subset of the values possible.
        Accordingly, it is appropriate for an agent to return a `badValue' response if a
        management station attempts to change this object to an inappropriate value."
    ::= { ip 1 }
```

```
ipDefaultTTL OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The default value inserted into the Time-To-Live field of the IP header of datagrams
        originated at this entity, whenever a TTL value is not supplied by the transport layer
        protocol."
    ::= { ip 2 }
```

```
ipInReceives OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of input datagrams received from interfaces, including those received
        in error."
    ::= { ip 3 }
```

```
ipInHdrErrors OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The number of input datagrams discarded due to errors in their IP headers, including
        bad checksums, version number mismatch, other format errors, time-to-live exceeded,
        errors discovered in processing their IP options, etc.”
    ::= { ip 4 }
```

```
ipInAddrErrors OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of input datagrams discarded because the IP address in their IP header's
        destination field was not a valid address to be received at this entity.  This count includes
        invalid addresses (e.g., 0.0.0.0) and addresses of unsupported Classes (e.g., Class E).  For
        entities which are not IP Gateways and therefore do not forward datagrams, this counter
        includes datagrams discarded because the destination address was not a local address."
    ::= { ip 5 }

ipForwDatagrams OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of input datagrams for which this entity was not their final IP destination,
        as a result of which an attempt was made to find a route to forward them to that final
        destination. In entities which do not act as IP Gateways, this counter will include only
        those packets which were Source-Routed via this entity, and the Source- Route option
        processing was successful."
    ::= { ip 6 }

ipInUnknownProtos OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of locally-addressed datagrams received successfully but discarded because
        of an unknown or unsupported protocol."
    ::= { ip 7 }

ipInDiscards OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of input IP datagrams for which no problems were encountered to prevent
        their continued processing, but which were discarded (e.g., for lack of buffer space).
        Note that this counter does not include any datagrams discarded while awaiting re-
        assembly."
    ::= { ip 8 }

ipInDelivers OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of input datagrams successfully delivered to IP user-protocols
        (including ICMP)."
    ::= { ip 9 }

ipOutRequests OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of IP datagrams which local IP user-protocols (including ICMP)
        supplied to IP in requests for transmission.  Note that this counter does not include any
        datagrams counted in ipForwDatagrams."
    ::= { ip 10 }

ipOutDiscards OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of output IP datagrams for which no problem was encountered to prevent
        their transmission to their destination, but which were discarded (e.g., for lack of buffer
        space).  Note that this counter would include datagrams counted in ipForwDatagrams if
        any such packets met this (discretionary) discard criterion."
    ::= { ip 11 }

ipOutNoRoutes OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        “The number of IP datagrams discarded because no route could be found to transmit
        them to their destination.  Note that this counter includes any packets counted in
        ipForwDatagrams which meet this `no-route’ criterion.  Note that this includes any
        datagarms which a host cannot route because all of its default gateways are down.”
    ::= { ip 12 }

ipReasmTimeout OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The maximum number of seconds which received fragments are held while they are
        awaiting reassembly at this entity.”
    ::= { ip 13 }

ipReasmReqds OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of IP fragments received which needed to be reassembled at this entity.”
    ::= { ip 14 }

ipReasmOKs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of IP datagrams successfully re- assembled.”
    ::= { ip 15 }

ipReasmFails OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of failures detected by the IP re- assembly algorithm (for whatever reason:
        timed out, errors, etc).  Note that this is not necessarily a count of discarded IP fragments
        since some algorithms (notably the algorithm in RFC 815) can lose track of the number
        of fragments by combining them as they are received."
    ::= { ip 16 }

ipFragOKs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of IP datagrams that have been successfully fragmented at this entity."
    ::= { ip 17 }

ipFragFails OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of IP datagrams that have been discarded because they needed to be
        fragmented at this entity but could not be, e.g., because their Don't Fragment flag was
        set."
    ::= { ip 18 }

ipFragCreates OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The number of IP datagram fragments that have been generated as a result of
        fragmentation at this entity."
    ::= { ip 19 }
```

the IP address tabletc "

the IP address table"§

The IP address table contains this entity’s IP addressing information.

```
ipAddrTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF IpAddrEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “The table of addressing information relevant to this entity’s IP addresses.”
    ::= { ip 20 }

ipAddrEntry OBJECT-TYPE
    SYNTAX              IpAddrEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “The addressing information for one of this entity’s IP addresses.”
    INDEX                   { ipAdEntAddr }
    ::= { ipAddrTable 1 }

IpAddrEntry ::=
    SEQUENCE {
        ipAdEntAddr
            IpAddress,
        ipAdEntIfIndex
            INTEGER,
        ipAdEntNetMask
            IpAddress,
        ipAdEntBcastAddr
            INTEGER,
        ipAdEntReasmMaxSize
            INTEGER (0..65535)
        }

ipAdEntAddr OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        “The IP address to which this entry’s addressing information pertains.”
    ::= { ipAddrEntry 1 }

ipAdEntIfIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The index value which uniquely identifies the interface to which this entry is applicable.
        The interface identified by a particular value of this index is the same interface as
        identified by the same value of ifIndex.”
    ::= { ipAddrEntry 2 }

ipAdEntNetMask OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The subnet mask associated with the IP address of this entry.  The value of the mask is
        an IP address with all the network bits set to 1 and all the hosts bits set to 0.”
    ::= { ipAddrEntry 3 }

ipAdEntBcastAddr OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The value of the least-significant bit in the IP broadcast address used for sending
        datagrams on the (logical) interface associated with the IP address of this entry.  For
        example, when the Internet standard all-ones broadcast address is used, the value will be
        1.  This value applies to both the subnet and network broadcasts addresses used by the
        entity on this (logical) interface.”
    ::= { ipAddrEntry 4 }

ipAdEntReasmMaxSize OBJECT-TYPE
    SYNTAX              INTEGER (0..65535)
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        “The size of the largest IP datagram which this entity can re-assemble from incoming IP
        fragmented datagrams received on this interface.”
    ::= { ipAddrEntry 5 }
```

the IP routing table

The IP routing table contains an entry for each route presently known to this entity.

```
ipRouteTable OBJECT-TYPE
    SYNTAX                  SEQUENCE OF IpRouteEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        “This entity’s IP Routing table.”
    ::= { ip 21 }
```

```
ipRouteEntry OBJECT-TYPE
    SYNTAX              IpRouteEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “A route to a particular destination.”
    INDEX                   { ipRouteDest }
    ::= { ipRouteTable 1 }

IpRouteEntry ::=
    SEQUENCE {
        ipRouteDest
            IpAddress,
        ipRouteIfIndex
            INTEGER,
        ipRouteMetric1
            INTEGER,
        ipRouteMetric2
            INTEGER,
        ipRouteMetric3
            INTEGER,
        ipRouteMetric4
            INTEGER,
        ipRouteNextHop
            IpAddress,
        ipRouteType
            INTEGER,
        ipRouteProto
            INTEGER,
        ipRouteAge
            INTEGER,
        ipRouteMask
            IpAddress,
        ipRouteMetric5
            INTEGER,
        ipRouteInfo
            OBJECT IDENTIFIER
        }

ipRouteDest OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-write
```

```
STATUS              mandatory
```

```
    DESCRIPTION
        "The destination IP address of this route.  An entry with a value of 0.0.0.0 is considered a
        default route.  Multiple routes to a single destination can appear in the table, but access to
        such multiple entries is dependent on the table- access mechanisms defined by the
        network management protocol in use."
    ::= { ipRouteEntry 1 }

ipRouteIfIndex OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        "The index value which uniquely identifies the local interface through which the next
        hop of this route should be reached.  The interface identified by a particular value of this
        index is the same interface as identified by the same value of ifIndex."
    ::= { ipRouteEntry 2 }

ipRouteMetric1 OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        "The primary routing metric for this route.  The semantics of this metric are determined
        by the routing-protocol specified in the route's ipRouteProto value.
        If this metric is not used, its value should be set to -1."
    ::= { ipRouteEntry 3 }

ipRouteMetric2 OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        "An alternate routing metric for this route.  The semantics of this metric are determined
        by the routing-protocol specified in the route's ipRouteProto value.
        If this metric is not used, its value should be set to -1."
    ::= { ipRouteEntry 4 }

ipRouteMetric3 OBJECT-TYPE
    SYNTAX                  INTEGER
```

```
ACCESS              read-write
STATUS              mandatory

DESCRIPTION
   “An alternate routing metric for this route.  The semantics of this metric are determined
   by the routing-protocol specified in the route’s ipRouteProto value.
   If this metric is not used, its value should be set to -1.”
::= { ipRouteEntry 5 }
```

```
ipRouteMetric4 OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “An alternate routing metric for this route.  The semantics of this metric are determined
        by the routing-protocol specified in the route’s ipRouteProto value.
        If this metric is not used, its value should be set to -1.”
    ::= { ipRouteEntry 6 }

ipRouteNextHop OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “The IP address of the next hop of this route. (In the case of a route bound to an interface
        which is realized via a broadcast media, the value of this field is the agent’s IP address on
        that interface.)”
    ::= { ipRouteEntry 7 }

ipRouteType OBJECT-TYPE
    SYNTAX              INTEGER
        {other(1),      none of the following
        invalid(2),     an invalidated route
                        route to directly
        direct(3),      connected (sub-)network
                        route to a non-local
        indirect(4)     host/network/sub-network
        }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “The type of route.  Note that the values direct(3) and indirect(4) refer to the notion of
        direct and indirect routing in the IP architecture. Setting this object to the value
        invalid(2) has the effect of invalidating the corresponding entry in the ipRouteTable
        object.  That is, it effectively dissasociates the destination identified with said entry from
        the route identified with said entry.  It is an implementation-specific matter as to whether
        the agent removes an invalidated entry from the table. Accordingly, management stations
        must be prepared to receive tabular information from agents that corresponds to entries
        not currently in use. Proper interpretation of such entries requires examination of the
        relevant ipRouteType object."
    ::= { ipRouteEntry 8 }
```

```
ipRouteProto OBJECT-TYPE
    SYNTAX                      INTEGER {
        other(1),               none of the following non-protocol information, e.g., manually
        local(2),               configured entries set via a network management protocol obtained
        netmgmt(3),             via ICMP, e.g., Redirect the remaining values are all gateway
routing
        icmp(4),                protocols
        egp(5),
        ggp(6),
        hello(7),
        rip(8),
        is-is(9),
        es-is(10),
        ciscoIgrp(11),
        bbnSpfIgp(12),
        ospf(13),
        bgp(14)
        }
    ACCESS                      read-only
    STATUS                      mandatory

    DESCRIPTION
        "The routing mechanism via which this route was learned.  Inclusion of values for
        gateway routing protocols is not intended to imply that hosts should support those
        protocols."
    ::= { ipRouteEntry 9 }

ipRouteAge OBJECT-TYPE
    SYNTAX                      INTEGER
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        "The number of seconds since this route was last updated or otherwise determined to be
        correct. Note that no semantics of `too old' can be implied except through knowledge of
        the routing protocol by which the route was learned."
    ::= { ipRouteEntry 10 }

ipRouteMask OBJECT-TYPE
    SYNTAX                      IpAddress
    ACCESS                      read-write
    STATUS                      mandatory
```

DESCRIPTION

“Indicate the mask to be logical-ANDed with the destination address before being compared to the value in the ipRouteDest field. For those systems that do not support arbitrary subnet masks, an agent constructs the value of the ipRouteMask by determining whether the value of the correspondent ipRouteDest field belong to a class-A, B, or C network, and then using one of:

```
        mask              network
        255.0.0.0         class-A
        255.255.0.0       class-B
        255.255.255.0     class-C


        If the value of the ipRouteDest is 0.0.0.0 (a default route), then the mask value is also
        0.0.0.0.  It should be noted that all IP routing subsystems implicitly use this mechanism."
    ::= { ipRouteEntry 11 }

ipRouteMetric5 OBJECT-TYPE
    SYNTAX                 INTEGER
    ACCESS                 read-write
    STATUS                 mandatory

    DESCRIPTION
        "An alternate routing metric for this route.  The semantics of this metric are determined
        by the routing-protocol specified in the route's ipRouteProto value.  If this metric is not
        used, its value should be set to -1."
    ::= { ipRouteEntry 12 }

ipRouteInfo OBJECT-TYPE
    SYNTAX                 OBJECT IDENTIFIER
    ACCESS                 read-only
    STATUS                 mandatory

    DESCRIPTION
        "A reference to MIB definitions specific to the particular routing protocol which is
        responsible for this route, as determined by the value specified in the route's
        ipRouteProto value.  If this information is not present, its value should be set to the
        OBJECT IDENTIFIER { 0 0 }, which is a syntatically valid object identifier, and any
        conformant implementation of ASN.1 and BER must be able to generate and recognize
        this value."
    ::= { ipRouteEntry 13 }
```

the IP Address Translation tabletc "the IP Address Translation table"§

The IP address translation table contain the IpAddress to 'physical' address equivalences. Some interfaces do not use translation tables for determining address equivalences (e.g., DDN-X.25 has an algorithmic method); if all interfaces are of this type, then the Address Translation table is empty, i.e., has zero entries.

```
ipNetToMediaTable OBJECT-TYPE
    SYNTAX                    SEQUENCE OF IpNetToMediaEntry
    ACCESS                    not-accessible
    STATUS                    mandatory

    DESCRIPTION
        “The IP Address Translation table used for mapping from IP addresses to physical
        addresses.”
    ::= { ip 22 }
```

```
ipNetToMediaEntry OBJECT-TYPE
    SYNTAX              IpNetToMediaEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        "Each entry contains one IpAddress to `physical' address equivalence."
    INDEX                   { ipNetToMediaIfIndex,ipNetToMediaNetAddress }
    ::= { ipNetToMediaTable 1 }

IpNetToMediaEntry ::=
    SEQUENCE {
        ipNetToMediaIfIndex
            INTEGER,
        ipNetToMediaPhysAddress
            PhysAddress,
        ipNetToMediaNetAddress
            IpAddress,
        ipNetToMediaType
            INTEGER }

ipNetToMediaIfIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The interface on which this entry's equivalence is effective.  The interface identified by
        a particular value of this index is the same interface as identified by the same value of
        ifIndex."
    ::= { ipNetToMediaEntry 1 }

ipNetToMediaPhysAddress OBJECT-TYPE
    SYNTAX              PhysAddress
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The media-dependent `physical' address."
    ::= { ipNetToMediaEntry 2 }

ipNetToMediaNetAddress OBJECT-TYPE
    SYNTAX              IpAddress
```

```
ACCESS              read-write
STATUS              mandatory

DESCRIPTION
   “The IpAddress corresponding to the media- dependent `physical’ address.”
::= { ipNetToMediaEntry 3 }
```

```
ipNetToMediaType OBJECT-TYPE
    SYNTAX                  INTEGER {
        other(1),           none of the following
        invalid(2),         an invalidated mapping
        dynamic(3),
        static(4) }
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        “The type of mapping. Setting this object to the value invalid(2) has the effect of
        invalidating the corresponding entry in the ipNetToMediaTable.  That is, it effectively
        dissasociates the interface identified with said entry from the mapping identified with
        said entry. It is an implementation-specific matter as to whether the agent removes an
        invalidated entry from the table.  Accordingly, management stations must be prepared to
        receive tabular information from agents that corresponds to entries not currently in use.
        Proper interpretation of such entries requires examination of the relevant
        ipNetToMediaType object.”
    ::= { ipNetToMediaEntry 4 }
```

additional IP objects

```
ipRoutingDiscards OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The number of routing entries which were chosen to be discarded even though they are
        valid.  One possible reason for discarding such an entry could be to free-up buffer space
        for other routing entries.”
    ::= { ip 23 }
```

ICMP grouptc "ICMP group"§

Implementation of the ICMP group is mandatory for all systems.

```
icmpInMsgs OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory
```

```
DESCRIPTION
    "The total number of ICMP messages which the entity received.  Note that this counter
    includes all those counted by icmpInErrors."
::= { icmp 1 }
```

```
icmpInErrors OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP messages which the entity received but determined as having
        ICMP-specific errors (bad ICMP checksums, bad length, etc.).”
    ::= { icmp 2 }

icmpInDestUnreachs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Destination Unreachable messages received.”
    ::= { icmp 3 }

icmpInTimeExcds OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Time Exceeded messages received.”
    ::= { icmp 4 }

icmpInParmProbs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Parameter Problem messages received.”
    ::= { icmp 5 }

icmpInSrcQuenchs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Source Quench messages received.”
    ::= { icmp 6 }

icmpInRedirects OBJECT-TYPE
    SYNTAX                   Counter
    ACCESS                   read-only
    STATUS                   mandatory

    DESCRIPTION
        “The number of ICMP Redirect messages received.”
    ::= { icmp 7 }
```

```
icmpInEchos OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Echo (request) messages received.”
    ::= { icmp 8 }

icmpInEchoReps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Echo Reply messages received.”
    ::= { icmp 9 }

icmpInTimestamps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Timestamp (request) messages received.”
    ::= { icmp 10 }

icmpInTimestampReps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Timestamp Reply messages received.”
    ::= { icmp 11 }

icmpInAddrMasks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Address Mask Request messages received.”
    ::= { icmp 12 }
```

```
icmpInAddrMaskReps OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The number of ICMP Address Mask Reply messages received.”
    ::= { icmp 13 }

icmpOutMsgs OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of ICMP messages which this entity attempted to send.  Note that this
        counter includes all those counted by icmpOutErrors.”
    ::= { icmp 14 }

icmpOutErrors OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The number of ICMP messages which this entity did not send due to problems
        discovered within ICMP such as a lack of buffers.  This value should not include errors
        discovered outside the ICMP layer such as the inability of IP to route the resultant
        datagram.  In some implementations there may be no types of error which contribute to
        this counter’s value.”
    ::= { icmp 15 }

icmpOutDestUnreachs OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The number of ICMP Destination Unreachable messages sent.”
    ::= { icmp 16 }
```

```
icmpOutTimeExcds OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Time Exceeded messages sent.”
    ::= { icmp 17 }

icmpOutParmProbs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Parameter Problem messages sent.”
    ::= { icmp 18 }

icmpOutSrcQuenchs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Source Quench messages sent.”
    ::= { icmp 19 }

icmpOutRedirects OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Redirect messages sent.  For a host, this object will always be
        zero, since hosts do not send redirects.”
    ::= { icmp 20 }

icmpOutEchos OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Echo (request) messages sent.”
    ::= { icmp 21 }

icmpOutEchoReps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Echo Reply messages sent.”
    ::= { icmp 22 }

icmpOutTimestamps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Timestamp (request) messages sent.”
    ::= { icmp 23 }

icmpOutTimestampReps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Timestamp Reply messages sent.”
    ::= { icmp 24 }

icmpOutAddrMasks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The number of ICMP Address Mask Request messages sent.”
    ::= { icmp 25 }

icmpOutAddrMaskReps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        "The number of ICMP Address Mask Reply messages sent."
    ::= { icmp 26 }
```

UDP grouptc "UDP group"§

Implementation of the UDP group is mandatory for all systems which implement the UDP.

```
udpInDatagrams OBJECT-TYPE
    SYNTAX          Counter
    ACCESS          read-only
    STATUS          mandatory

    DESCRIPTION
        "The total number of UDP datagrams delivered to UDP users."
    ::= { udp 1 }

udpNoPorts OBJECT-TYPE
    SYNTAX          Counter
    ACCESS          read-only
    STATUS          mandatory

    DESCRIPTION
        "The total number of received UDP datagrams for which there was no application at the
        destination port."
    ::= { udp 2 }

udpInErrors OBJECT-TYPE
    SYNTAX          Counter
    ACCESS          read-only
    STATUS          mandatory

    DESCRIPTION
        "The number of received UDP datagrams that could not be delivered for reasons other
        than the lack of an application at the destination port."
    ::= { udp 3 }

udpOutDatagrams OBJECT-TYPE
    SYNTAX          Counter
    ACCESS          read-only
    STATUS          mandatory

    DESCRIPTION
```

```
        "The total number of UDP datagrams sent from this entity."
    ::= { udp 4 }
```

the UDP Listener tabletc "the UDP Listener table"§

The UDP listener table contains information about this entity’s UDP end-points on which a local application is currently accepting datagrams.

```
udpTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF UdpEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “A table containing UDP listener information.”
    ::= { udp 5 }

udpEntry OBJECT-TYPE
    SYNTAX              UdpEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “Information about a particular current UDP listener.”
    INDEX                   { udpLocalAddress, udpLocalPort }
    ::= { udpTable 1 }

UdpEntry ::=
    SEQUENCE {
        udpLocalAddress
            IpAddress,
        udpLocalPort
            INTEGER (0..65535)
    }

udpLocalAddress OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The local IP address for this UDP listener.  In the case of a UDP listener which is
        willing to accept datagrams for any IP interface associated with the node, the value
        0.0.0.0 is used.”
    ::= { udpEntry 1 }
```

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

    DESCRIPTION
        “The local port number for this UDP listener.”
    ::= { udpEntry 2 }
```

Transmission grouptc "

Transmission group"§

Based on the transmission media underlying each interface on a system, the corresponding portion of the Transmission group is mandatory for that system. When Internet-standard definitions for managing transmission media are defined, the transmission group is used to provide a prefix for the names of those objects. Typically, such definitions reside in the experimental portion of the MIB until they are "proven", then as a part of the Internet standardization process, the definitions are accordingly elevated and a new object identifier, under the transmission group is defined. By convention, the name assigned is:

```
    type OBJECT IDENTIFIER  ::= { transmission number }
```

where "type" is the symbolic value used for the media in the ifType column of the ifTable object, and "number" is the actual integer value corresponding to the symbol.

SNMP grouptc "SNMP group"§

Implementation of the SNMP group is mandatory for all systems which support an SNMP protocol entity. Some of the objects defined below will be zero-valued in those SNMP implementations that are optimized to support only those functions specific to either a management agent or a management station. In particular, it should be observed that the objects below refer to an SNMP entity, and there may be several SNMP entities residing on a managed node (e.g., if the node is hosting acting as a management station).

```
snmpInPkts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of Messages delivered to the SNMP entity from the transport service."
    ::= { snmp 1 }

snmpOutPkts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
```

```
        “The total number of SNMP Messages which were passed from the SNMP protocol
        entity to the transport service.”
    ::= { snmp 2 }

snmpInBadVersions OBJECT-TYPE
    SYNTAX                    Counter
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        “The total number of SNMP Messages which were delivered to the SNMP protocol
        entity and were for an unsupported SNMP version.”
    ::= { snmp 3 }

snmpInBadCommunityNames OBJECT-TYPE
    SYNTAX                    Counter
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        “The total number of SNMP Messages delivered to the SNMP protocol entity which used
        a SNMP community name not known to said entity.”
    ::= { snmp 4 }

snmpInBadCommunityUses OBJECT-TYPE
    SYNTAX                    Counter
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        “The total number of SNMP Messages delivered to the SNMP protocol entity which
        represented an SNMP operation which was not allowed by the SNMP community named
        in the Message.”
    ::= { snmp 5 }

snmpInASNParseErrs OBJECT-TYPE
    SYNTAX                    Counter
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        “The total number of ASN.1 or BER errors encountered by the SNMP protocol entity
        when decoding received SNMP Messages.”
    ::= { snmp 6 }

{ snmp 7 } is not used

snmpInTooBigs OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of SNMP PDUs which were delivered to the SNMP protocol entity
        and for which the value of the error-status field is `tooBig’.”
    ::= { snmp 8 }

snmpInNoSuchNames OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of SNMP PDUs which were delivered to the SNMP protocol entity
        and for which the value of the error-status field is `noSuchName’.”
    ::= { snmp 9 }

snmpInBadValues OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of SNMP PDUs which were delivered to the SNMP protocol entity
        and for which the value of the error-status field is `badValue’.”
    ::= { snmp 10 }

snmpInReadOnlys OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number valid SNMP PDUs which were delivered to the SNMP protocol entity
        and for which the value of the error-status field is `readOnly’.  It should be noted that it
        is a protocol error to generate an SNMP PDU which contains the value `readOnly' in the
        error-status field, as such this object is provided as a means of detecting incorrect
        implementations of the SNMP."
    ::= { snmp 11 }

snmpInGenErrs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of SNMP PDUs which were delivered to the SNMP protocol entity
        and for which the value of the error-status field is `genErr'."
    ::= { snmp 12 }

snmpInTotalReqVars OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of MIB objects which have been retrieved successfully by the SNMP
        protocol entity as the result of receiving valid SNMP Get-Request and Get-Next PDUs."
    ::= { snmp 13 }

snmpInTotalSetVars OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of MIB objects which have been altered successfully by the SNMP
        protocol entity as the result of receiving valid SNMP Set-Request PDUs."
    ::= { snmp 14 }

snmpInGetRequests OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of SNMP Get-Request PDUs which have been accepted and processed
        by the SNMP protocol entity."
    ::= { snmp 15 }

snmpInGetNexts OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of SNMP Get-Next PDUs which have been accepted and processed by
        the SNMP protocol entity.”
    ::= { snmp 16 }

snmpInSetRequests OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of SNMP Set-Request PDUs which have been accepted and processed
        by the SNMP protocol entity.”
    ::= { snmp 17 }

snmpInGetResponses OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of SNMP Get-Response PDUs which have been accepted and
        processed by the SNMP protocol entity.”
    ::= { snmp 18 }

snmpInTraps OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        “The total number of SNMP Trap PDUs which have been accepted and processed by the
        SNMP protocol entity.”
    ::= { snmp 19 }
```

```
snmpOutTooBigs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of SNMP PDUs which were generated by the SNMP protocol entity
        and for which the value of the error-status field is `tooBig.’”
    ::= { snmp 20 }

snmpOutNoSuchNames OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of SNMP PDUs which were generated by the SNMP protocol entity
        and for which the value of the error-status is `noSuchName’.”
    ::= { snmp 21 }

snmpOutBadValues OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of SNMP PDUs which were generated by the SNMP protocol entity
        and for which the value of the error-status field is `badValue’.”
    ::= { snmp 22 }

{ snmp 23 } is not used

snmpOutGenErrs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “The total number of SNMP PDUs which were generated by the SNMP protocol entity
        and for which the value of the error-status field is `genErr’.”
    ::= { snmp 24 }

snmpOutGetRequests OBJECT-TYPE
```

```
SYNTAX          Counter
ACCESS          read-only
STATUS          mandatory

DESCRIPTION
    “The total number of SNMP Get-Request PDUs which have been generated by the
    SNMP protocol entity.”
::= { snmp 25 }
```

```
snmpOutGetNexts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of SNMP Get-Next PDUs which have been generated by the SNMP
        protocol entity."
    ::= { snmp 26 }

snmpOutSetRequests OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of SNMP Set-Request PDUs which have been generated by the SNMP
        protocol entity."
    ::= { snmp 27 }

snmpOutGetResponses OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of SNMP Get-Response PDUs which have been generated by the
        SNMP protocol entity."
    ::= { snmp 28 }

snmpOutTraps OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        "The total number of SNMP Trap PDUs which have been generated by the SNMP
        protocol entity."
    ::= { snmp 29 }

snmpEnableAuthenTraps OBJECT-TYPE
    SYNTAX              INTEGER { enabled(1), disabled(2) }
    ACCESS              read-write
```

```
STATUS                 mandatory

DESCRIPTION
   “Indicates whether the SNMP agent process is permitted to generate authentication-
   failure traps.  The value of this object overrides any configuration information; as such, it
   provides a means whereby all authentication-failure traps may be disabled. Note that it is
   strongly recommended that this object be stored in non-volatile memory so that it
   remains constant between re-initializations of the network management system.”
::= { snmp 30 }
```

Station Grouptc "

Station Group"§

```
stationTime OBJECT-TYPE
    SYNTAX                     OCTET STRING (SIZE(13))
    ACCESS                     read-write
    STATUS                     mandatory

    DESCRIPTION
        “The station time is the time as perceived by the station. The string returned is a subset of
        UTC Time as specified in the Draft Recommendation X.409 Message Handling
        Systems : Presentation Transfer Syntax and Notation (ASN 1).The formats acceptable
        when the time is being set by a manager are :

        a. 11 octets
            YYMMDDhhmmZ

        b. 13 octets
            YYMMDDhhmmssZ
            where
            YY are two digits representing the year,
            MM are two digits representing the month
            DD are two digits representing the day
            hh are two digits representing the hour
            mm are two digits representing the month
            ss are two digits representing the seconds
            Z is the ASCII character Z.


        This time returned by the station in response to a Get or GetNext request will always be
        the 13 octet variant.”
    ::=  { station 1 }

stationResetCounter OBJECT-TYPE
    SYNTAX                     Counter
    ACCESS                     read-only
    STATUS                     mandatory

    DESCRIPTION
        ”This parameter is a count of the number of times the station has been reset since the last
        station intialisation.”
    ::=  { station 2 }
```

```
freeBufferCount OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This parameter represents the number of free buffers remaining in this station.”
    ::=  { station 3 }
```

```
freeHeaderCount OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter represents the number of free headers remaining in this station.”
    ::=  { station 4 }

physBlkSize OBJECT-TYPE
    SYNTAX              INTEGER (300..1600)
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter represents the current size in bytes of the physical block used by the
        station’s buffer manager.”
    ::=  { station 5 }

newPhysBlkSize OBJECT-TYPE
    SYNTAX              INTEGER (300..1600)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This parameter represents the size in bytes of the physical block, that will be used by the
        station’s buffer manager, after the next station reset.”
    ::=  { station 6 }

resetStation OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object is used to reset the station. This action does not clear any configured data, so
        that after the reset, the box does not return to the factory defaults.Only two values are
        valid to use with this object. A value of 1 is always returned when a GET is performed
        upon the object. To force a reset,  a value of 2 must be used in the SET.  A SET with a
        value of 1 has no effect. “

    ::=  { station 7 }
```

```
initStation OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object is used to initialise the station. Any configured data is lost, and all data
        values are returned to factory default values.Only two values are valid to use with this
        object. A value of 1 is always returned when a GET is performed upon the object. To
        force the initialisation,  a value of 2 must be used in the SET.  A SET with a value of 1
        has no effect.”
    ::=  { station 8 }
```

```
resetStats OBJECT-TYPE
    SYNTAX                 INTEGER
    ACCESS                 read-write
    STATUS                 mandatory

    DESCRIPTION
        ”The statistics counters are reset to zero by performing a set on this object. This reset
        occurs immediately. See the Statistics Objects section for a list of these counters.Only
        two values are valid to use with this object. A value of 1 is always returned when a GET
        is performed upon the object. To force a reset,  a value of 2 must be used in the SET.  A
        SET with a value of 1 has no effect.”
    ::=  { station 9 }

procLoadTable OBJECT-TYPE
    SYNTAX                 SEQUENCE OF ProcLoadValue
    ACCESS                 not-accessible
    STATUS                 mandatory

    DESCRIPTION
        ”List of processor load statistics.”
    ::=  { station 10 }

ProcLoadValue ::=
    SEQUENCE {
        procLoadTable
            INTEGER
    }

processorLoading OBJECT-TYPE
    SYNTAX                 INTEGER
    ACCESS                 read-only
    STATUS                 mandatory

    DESCRIPTION
        ”Each instance of this object gives the percentage loading on a particular processor.
        Currently the first instance gives the percentage loading on the station’s main processor
        and the second instance gives the loading on the station’s sub-processor.”
    ::=  { procLoadTable 1 }

trapDestinationTable OBJECT IDENTIFIER ::= { station 11 }
```

The trapDestinationTable object is used only as a place holder. It is replaced by the s3agentTrapRcvr stuff.

```
authenticationTrapStatus OBJECT-TYPE
    SYNTAX              INTEGER (0..1)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object determines if the station will send a Trap upon detection of an authentication
        failure error.”
    ::=  { station 14 }
```

```
serialTxQueueSize OBJECT-TYPE
    SYNTAX                  INTEGER (1..127)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This parameter indicates the maximum number of LAPB frames that can be enqueued,
        waiting for transmission over the serial interface.”
    ::=  { station 15 }

internalQueueCurrentLength OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the length of the enqueued information specific to the internal operation of the
        bridge.”
    ::=  { station 16 }

queueUpperLimit OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This is the upper limit of the enqueued information, for all the queues, specific to the
        operation of the bridge.”
    ::=  { station 17 }

lanQueueSize OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This variable reports the maximum number of frames that are enqueued waiting for
        transmission over the LAN interface.”
    ::=  { station 18 }
```

8023 group

```
ieee8023Table OBJECT-TYPE
    SYNTAX              SEQUENCE OF Ieee8023Entry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”A list of IEEE 802.3 Interface entries.”
    ::= { ieee8023 1 }
```

```
ieee8023Entry OBJECT-TYPE
    SYNTAXIeee8023Entry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION “”
    INDEX                       { ieee8023Index }
    ::=  { ieee8023Table 1 }

Ieee8023Entry ::= SEQUENCE {
    ieee8023Index
        INTEGER,
    ieee8023FramesTransmittedOks
        Counter,
    ieee8023SingleCollisionFrames
        Counter,
    ieee8023MultipleCollisionFrames
        Counter,
    ieee8023OctetsTransmittedOks
        Counter,
    ieee8023DeferredTransmissions
        Counter,
    ieee8023MulticastFramesTransmittedOks
        Counter,
    ieee8023BroadcastFramesTransmittedOks
        Counter,
    ieee8023Latecollisions
        Counter,
    ieee8023ExcessiveCollisionsColls
        Counter,
    ieee8023InternalMacTransmitErrors
        Counter,
    ieee8023CarrierSenseErrors
        Counter,
    ieee8023ExcessiveDeferals
        Counter,
    ieee8023FramesReceivedOks
        Counter,
    ieee8023OctetsReceivedOks
        Counter,
    ieee8023MulticastFramesReceivedOks
        Counter,
ieee8023BroadcastFramesReceivedOks
    Counter,
ieee8023FrameTooLongs
    Counter,
ieee8023AlignmentErrors
    Counter,
    ieee8023FCSErrors
        Counter,
    ieee8023inRangeLengthErrors
        Counter,
    ieee8023outOfRangeLengthFields
        Counter,
    ieee8023InternalMACReceiveErrors
        Counter,
    ieee8023InitializeMAC
        INTEGER,
    ieee8023PromiscuousReceiveStatus
        INTEGER,
    ieee8023MACSubLayerStatus
        INTEGER,
    ieee8023TransmitStatus
        INTEGER,
    ieee8023MulticastReceiveStatus
        INTEGER,
    ieee8023MACAddress
        OCTET STRING,
    ieee8023SQETestErrors
        Counter
    }

ieee8023FramesTransmittedOks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames that are successfully transmitted. This counter is
        incremented when a frame is successfully transmitted, even if one of the collision
        Counters (following two objects) are also counted. In other words, this count includes
        transmissions that succeeded after some number of collisions. This count includes
        multicast and broadcast frames. “
    ::= { ieee8023Entry 2 }

ieee8023SingleCollisionFrames OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory
```

```
DESCRIPTION
    ”This contains a count of frames that are involved in a single collision, and are
    subsequently transmitted successfully. If this counter is incremented then the
    ieee8023FramesTransmittedOKs count will also be incremented AND
    ieee8023MultipleCollisionFrames will NOT be incremented.”
::= { ieee8023Entry 3 }
```

```
ieee8023MultipleCollisionFrames OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames that are involved in more than one collision and are
        subsequently transmitted successfully. If this counter is incremented then the
        ieee8023FramesTransmittedOKs count will also be incremented AND
        ieee8023SingleCollisionFrames will NOT be incremented.”
    ::= { ieee8023Entry 4 }

ieee8023OctetsTransmittedOks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of data and padding octets of frames that are successfully
        transmitted. This counter includes octets in broadcast and multicast frames that are
        transmitted. This count does not include the address fields, length/type field and CRC
        field of the frame. In Ethernet-Speak, it is JUST the data field (IP header, etc).”
    ::= { ieee8023Entry 5 }

ieee8023DeferredTransmissions OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames whose transmission was delayed on its first attempt
        because the medium was busy. Frames involved in collisions are NOT included in this
        count.”
    ::= { ieee8023Entry 6 }

ieee8023MulticastFramesTransmittedOks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames successfully transmitted to a group destination
        (multicast) address other than broadcast.”
    ::= { ieee8023Entry 7 }

ieee8023BroadcastFramesTransmittedOks OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of frames successfully transmitted to a group destination
        (multicast) address other than broadcast.”
    ::= { ieee8023Entry 8 }
```

```
ieee8023LateCollisions OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of frames successfully transmitted to the broadcast address. This
        does not include frames sent to a multicast address.”
    ::= { ieee8023Entry 9 }

ieee8023ExcessiveCollisions OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of times that a collision has been detected later than 512 bit-times
        into the transmission of a packet. 512 bit-times is 51.2 micro-seconds. This type
        ofcollision is counted twice: both as a late collision and as a regular, run of the mill
        collision.”
    ::= { ieee8023Entry 10 }

ieee8023InternalMACTransmitErrors OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of frames that due to excessive collisions are not transmitted
        successfully.”
    ::= { ieee8023Entry 11 }

ieee8023CarrierSenseErrors OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of frames which would otherwise be transmitted by the station,
        but could not be sent due to an internal MAC sublayer transmit error. If this counter
        isincremented then none of the other Counters in this section are incremented. The exact
        meaning and mechanism for incrementing this counter is implementation dependent. In
        short, this is the count of errors for which there are no specific Counters.”
    ::= { ieee8023Entry 12 }

ieee8023ExcessiveDeferrals OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of frames which deferred for an excessive period of time.”
    ::= { ieee8023Entry 13 }
```

```
ieee8023FramesReceivedOks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames that are successfully received. This does not include
        frames received with a frame-too-long, FCS, length, or alignment errors or frames lost
        due to internal MAC sublayer error. This counter includes broadcast and multicast frames
        that are received.”
    ::= { ieee8023Entry 14 }

ieee8023OctetsReceivedOks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of data and padding octets in frames that are successfully
        received. This does not include frames received with a frame-too-long, FCS, length, or
        alignment errors or frames lost due to internal MAC sublayer error. This counter includes
        octets in broadcast and multicast frames that are received. This counter does not include
        the octets in the address, length/type and CRC fields.”
    ::= { ieee8023Entry 15 }

ieee8023MulticastFramesReceivedOks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames that are successfully received and are directed to an
        active non-broadcast group address. This does  not include frames received with a frame-
        too-long, FCS, length, or alignment errors or frames lost due to internal MAC sublayer
        error. Broadcast packets are not included.”
    ::= { ieee8023Entry 16 }

ieee8023BroadcastFramesReceivedOks OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        ”This contains a count of frames that are successfully received and are directed to the
        broadcast group address. This does not include frames received with a frame-too- long,
        FCS, length, or alignment errors or frames lost due to internal MAC sublayer error.
        Multicast packets are not included.”
    ::= { ieee8023Entry 17 }

ieee8023FrameTooLongs OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames that are received and exceed the maximum permitted
        frame size.”
    ::= { ieee8023Entry 18 }

ieee8023AlignmentErrors OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames that are not an integral number of octets in length and
        do not pass the FCS check.”
    ::= { ieee8023Entry 19 }

ieee8023FCSErrors OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This contains a count of frames that are an integral number of octets in length that do
        not pass the FCS check.”
    ::= { ieee8023Entry 20 }

ieee8023inRangeLengthErrors OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
```

```
        ”This contains a count of frames with a length field value between the minimum
        unpadded LLC data size and the maximum allowed LLC data size inclusive, and does not
        match the number of LLC data octets received. The counter also contains frames with a
        length field value less than the minimum unpadded LLC data size.”
    ::= { ieee8023Entry 21 }

ieee8023outOfRangeLengthFields OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of frames with a length field value greater than the maximum
        allowed LLC data size.”
    ::= { ieee8023Entry 22 }

ieee8023InternalMACReceiveErrors OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This contains a count of frames which would otherwise be received by the station, but
        could not be accepted due to an internal MAC sublayer receive error. If this counter is
        incremented then none of the other Counters in this section are incremented. The exact
        meaning and mechanism for incrementing this counter is implementation dependent. In
        short, this is the count of errors for which there are no specific Counters.”
    ::= { ieee8023Entry 23 }

ieee8023InitializeMAC OBJECT-TYPE
    SYNTAX                  INTEGER {
        initialize (1)
        }
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”Resets the MAC layer counters. When set, only the value initialize may be used. When
        read, this variable always returns the most recent value to which ieee8023InitializeMAC
        was set to. At the moment, this function solely serves to zero the MAC counters.”
    ::= { ieee8023Entry 24 }
```

```
ieee8023PromiscuousReceiveStatus OBJECT-TYPE
    SYNTAX              INTEGER {
        enabled (1)
        }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This variable causes the MAC sublayer to enter promiscuous receive state.”
    ::= { ieee8023Entry 25 }

ieee8023MACSubLayerStatus OBJECT-TYPE
    SYNTAX              INTEGER {
        enabled (1)
        }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”Causes the MAC sublayer to enter the normal operational state at idle.”
    ::= { ieee8023Entry 26 }

ieee8023TransmitStatus OBJECT-TYPE
    SYNTAX              INTEGER {
        enabled (1),
        disabled (2)
        }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This variable allows the MAC sublayer to be explicitly enabled/disabled for transmitting
        frames.”
    ::= { ieee8023Entry 27 }

ieee8023MulticastReceiveStatus OBJECT-TYPE
    SYNTAX              INTEGER {
        enabled (1)
        }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
```

```
        ”Enables reception of multicast frames.”
    ::= { ieee8023Entry 28 }

ieee8023MACAddress OBJECT-TYPE
    SYNTAX                    OCTET STRING (SIZE (6))
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This object contains the MAC station address (i.e. the Ethernet address).”
    ::= { ieee8023Entry 29 }

ieee8023SQETestErrors OBJECT-TYPE
    SYNTAX                    COUNTER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This contains a count of times that the SQE TEST ERROR status was received.”
    ::= { ieee8023Entry 30 }
```

Physical Serial Grouptc "Physical Serial Group"§

```
phySerIfNumber OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”The number of Physical Serial interfaces present on this system.”
    ::=  { phySerIf 1 }

phySerIfTable OBJECT-TYPE
    SYNTAX                    SEQUENCE OF PhySerIfEntry
    ACCESS                    not-accessible
    STATUS                    mandatory

    DESCRIPTION
        ”A list of interface entries. The number of entries is given by the value of
        phySerIfNumber.”
    ::=  { phySerIf 2 }
```

```
phySerIfEntry OBJECT-TYPE
    SYNTAXPhySerIfEntry
    ACCESS                  not-accessible
    STATUS                  mandatory
    INDEX                       { phySerIfIndex }
    ::=  { phySerIfTable 1 }

PhySerIfEntry ::= SEQUENCE {
    phySerIfIndex
        INTEGER,
    phySerIfInterfaceType
        INTEGER,
    phySerIfMeasuredSpeed
        INTEGER,
    phySerIfIsSpeedsettable
        INTEGER,
    phySerIfPortSpeed
        INTEGER,
    phySerIfTransitDelay
        INTEGER,
    phySerIfT1clockSource
        INTEGER,
    phySerIfT1SlotLvalue
        INTEGER,
    phySerIfT1SlotHvalue
        INTEGER,
    phySerIfT1dRatePerChan
        INTEGER,
    phySerIfT1frameAndCode
        INTEGER,
    phySerIfMeasuredSpeed
        INTEGER,
    phySerIfPartnerAddress
        OCTET STRING (SIZE (6))
    }

phySerIfIndex OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The interface on which this entry’s equivalence is effective.”
```

```
::=  { phySerIfEntry 1 }
```

```
phySerIfInterfaceType OBJECT-TYPE
    SYNTAX              INTEGER {
        x21dte(1),
        x21dce(2),
        rs449(3),
        g703(4),
        v35(5),
        v35btb(6),
        rs232(7),
        t1(8)
        }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”Each of the values above indicates a particular personality module and hence the
        physical interface present. It is only possible to modify certain values of this variable, as
        follows:

        The value x21dte(1) may be modified to x21dce(2), and vice versa.


        The value v35(5) may be modified to v35btb(6), and vice versa. The changes will occur
        after the next link reset.”
    ::=  { phySerIfEntry 2 }

phySerIfMeasuredSpeed OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter is the value of the measured channel speed in bits per second.”
    ::=  { phySerIfEntry 3 }

phySerIfIsSpeedsettable OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter indicates whether the speed is settable or not for the particular
        personality module. This parameter does not guarantee that this will be the value of the
        clock used over the serial link."
    ::=  { physSerIfEntry 4 }

physSerIfPortSpeed OBJECT-TYPE
    SYNTAX                      INTEGER {
        b1200(1200),
        b2400(2400),
        b4800(4800),
        b9600(9600),
        b19200(19200),
        b24000(24000),
        b32000(32000),
        b48000(48000),
        b64000(64000),
        b256000(256000),
        b512000(512000),
        b1024000(1024000),
        b2048000(2048000)
        }
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        "The port speed is the value of the personality module sourced clock. It may be accessed
        if the value of isSpeedSettable is 1. In many cases a special cable is needed for the
        parameter to affect station operation - see product specifications for details.For the
        RS232 personality module the port speed can be set to any of the values 2400 through
        48000 listed above. For all the other personality modules for which port speed is a
        settable parameter, the values 19600 and 24000 are invalid.  These modules can assume
        any of the other speeds, dependent upon the product type - see product specifications for
        details.When a change has been made to this object, the new value will become effective
        after the next link reset."
    ::=  { physSerIfEntry 5 }

physSerIfTransitDelay OBJECT-TYPE
    SYNTAX                      INTEGER
    ACCESS                      read-only
    STATUS                      mandatory

    DESCRIPTION
        "This parameter indicates the time taken for a frame to traverse the physSer link between
        the local and remote station. Its value is automatically determined by the station's
        software.”
    ::=  { phySerIfEntry 6 }

phySerIfT1clockSource OBJECT-TYPE
    SYNTAX                      INTEGER(1..3)
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”This parameter indicates from where the clock is being derived. If its value is 1 then the
        clock signal comes from the data in the frame being received, if it is 2 then the clock is
        derived from an internal source and if its value is 3 the clock source is external. This
        parameter is only relevant for the T1 personality module. When a change has been made
        to this object, the new value will become effective after the next link reset.”
    ::=  { phySerIfEntry 7 }
```

```
phySerIfT1SlotLvalue OBJECT-TYPE
    SYNTAX              INTEGER (1..24)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the  value of the lower time slot chosen within a frame of the 1.544MHz T1
        service. This parameter is only relevant for the T1 personality module. When a change
        has been made to this object, the new value will become effective after the next link
        reset.”
    ::=  { phySerIfEntry 8 }

phySerIfT1SlotHvalue OBJECT-TYPE
    SYNTAX              INTEGER (1..24)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the  value of the higher time slot chosen within a frame of the 1.544MHz T1
        service. This parameter is only relevant for the T1 personality module. When a change
        has been made to this object the new value, will become effective after the next link
        reset.”
    ::=  { phySerIfEntry 9 }

phySerIfT1dRatePerChan OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object can have either of two values depending on the data rate required. If its
        value is 1 a data rate of 64k is provided, if its value is 5 then the data rate provided is
        56k. This parameter is only relevant for the T1 personality module. When a change has
        been made to this object, the new value will become effective after the next link reset.”
    ::=  { phySerIfEntry 10 }

phySerIfT1frameAndCode OBJECT-TYPE
    SYNTAX              INTEGER (1..4)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
```

```
        ”There are four possible values of frame and code. These values represent the encoding
        on the 1.544MHz T1 service. The values and the codes they represent are as follows.

        fcD4AndB8zs    1
        fcD4Only       2
        fcEsfAndB8zs   3
        fcEsfOnly      4


        This parameter is only relevant for the T1 personality module. When a change has been
        made to this object, the new value will become effective after the next link reset.”
    ::=  { physSerIfEntry 11 }

phySerIfpartnerAddress OBJECT-TYPE
    SYNTAX                  OCTET STRING (SIZE (0..6))
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The mac address of the remote station to which this particular serial link is connected.”
    ::=  { physSerIfEntry 12 }
```

Multilink Grouptc "Multilink Group"§

```
mlinkNumber OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The number of multilink interfaces, (regardless of their current state) present on this
        system.”
    ::=  { mlink 1 }

mlinkTable OBJECT-TYPE
    SYNTAX                  SEQUENCE OF MlinkEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        ”A list of multilink entries. The number of entries is given by the value of
        mlinkNumber.”
    ::=  { mlink 2 }

mlinkEntry OBJECT-TYPE
    SYNTAX                    MlinkEntry
    ACCESS                    not-accessible
    STATUS                    mandatory
    INDEX                         { mlinkIndex }
    ::=  { mlinkTable 1 }

MlinkEntry ::= SEQUENCE {
    mlinkIndex
        INTEGER,
    mlinkState
        INTEGER,
    mlinkSendSeq
        INTEGER,
    mlinkRcvSeq
        INTEGER,
    mlinkSendUpperEdge
        INTEGER,
    mlinkRcvUpperEdge
        INTEGER,
    mlinkFramesLost
        Counter,
    deletedMlinkFrames
        Counter,
    expressQueueCurrentLength
        INTEGER,
    expressQueueUpperLimit
        INTEGER,
    hiPriQueueCurrentLength
        INTEGER,
    hiPriQueueUpperLimit
        INTEGER,
    loPriQueueCurrentLength
        INTEGER,
    loPriQueueUpperLimit
        INTEGER,
    mlinkChanCompression
        INTEGER,
    mlinkChanCompressionRatio
        OCTET STRING
    }

mlinkIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This value reports the particular instance of multi-link being referenced.”
    ::=  { mlinkEntry 1 }

mlinkState OBJECT-TYPE
    SYNTAX              INTEGER (1..4)
    ACCESS              read-only
    STATUS              mandatory
```

DESCRIPTION

”This parameter indicates the current state of the multi link connection. The states and corresponding codes are defined as follows.

```
        negotiating       1
        notInUse          2
        inUseActive       3
        inUseInactive     4"

    ::=  { mlinkEntry 2 }

mlinkSendSeq OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This parameter indicates the current state of the multi link connection. The states and
        corresponding codes are defined as follows.

        negotiating       1
        notInUse          2
        inUseActive       3
        inUseInactive     4"

    ::=  { mlinkEntry 3 }

mlinkRcvSeq OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This parameter represents the sequence number of the next-in-sequence multilink frame
        to be received.”
    ::=  { mlinkEntry 4 }

mlinkSendUpperEdge OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This parameter is the sum of the value of the last received acknowledgement and the
        value of the multilink window size.”
    ::=  { mlinkEntry 5 }
```

```
mlinkRcvUpperEdge OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter is the sum of the value of the sequence number sent in the last
        acknowledgement and the receive window size.”
    ::=  { mlinkEntry 6 }
```

```
mlinkFramesLost OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This parameter represents the number of multilink packets lost because the out of
        sequence queue was full or the time spent waiting for an out of sequence frame (in order
        to re-establish the sequence) exceeded some implementation specific period. It is the
        count since the last statistics reset of station restart.”
    ::=  { mlinkEntry 7 }

deletedMlinkFrames OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the number of multilink frames which have been deleted, because they arrived
        out of sequence and were outside of the multi-link window.”
    ::=  { mlinkEntry 8 }
expressQueueCurrentLength OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the number of items enqueued in the multilink express queue.”
    ::=  { mlinkEntry 9 }

expressQueueUpperLimit OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the upper limit of the enqueued information specific to the multilink express
        queue.”
    ::=  { mlinkEntry 10 }

hiPriQueueCurrentLength OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the number of items enqueued in the multilink high priority queue.”
    ::=  { mlinkEntry 11 }

hiPriQueueUpperLimit OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the upper limit of the enqueued information specific to the multilink high
        priority queue.”
    ::=  { mlinkEntry 12 }

loPriQueueCurrentLength OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the number of items enqueued in the multilink low priority queue.”
    ::=  { mlinkEntry 13 }

loPriQueueUpperLimit OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the upper limit of the enqueued information specific to the multilink low priority
        queue.”
    ::=  { mlinkEntry 14 }
mlinkChanCompression OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “Enable and disable serial line compression, can take the following values :

        compression enabled  1
```

```
          compression disabled  2"

     ::= { mlinkEntry 16 }

mlinkChanCompressionRatio OBJECT-TYPE
     SYNTAX                    OCTET STRING
     ACCESS                    read-only
     STATUS                    mandatory

     DESCRIPTION
          "An octet string which, when compression is enabled returns the rateof compression
          acheived on this link.  An octet string is returned which indicates for each byte of
          compressed data transmitted how many uncompressed byteswhere processed.  The string
          will be in the form of a  decimal string with up to two decimal places. i.e. as follows :


          '1.23' - indicates a compression ration of 1.23 uncompressed data bytes to each
          compressed byte."

     ::= { mlinkEntry 17 }
```

```
mlinkWindow OBJECT-TYPE
    SYNTAX                INTEGER
    ACCESS                read-write
    STATUS                mandatory

    DESCRIPTION
        ”The Multilink window size parameter defines the number of sequentially numbered
        multilink frames that the DCE or DTE may transfer to its SLP beyond the lowest
        numbered multilink frame which has not yet been acknowledged. This parameter is not
        instanced; it affects all instances of multilink.”
    ::=  { mlink 3 }

mlinkRxTimeout OBJECT-TYPE
    SYNTAX                INTEGER
    ACCESS                read-write
    STATUS                mandatory

    DESCRIPTION
        ”The time, in 50ms units, after which the multilink resets due to a receive sequence error.
        This parameter is not instanced; it affects all instances of multilink.”
    ::=  { mlink 4 }
```

Bridge Grouptc "Bridge Group"§

```
portNumber OBJECT-TYPE
    SYNTAX                INTEGER
    ACCESS                read-only
    STATUS                mandatory

    DESCRIPTION
        ”The number of ports supported by the bridging function in this station.”
    ::=  { bridge 1 }

bridgeStatsTable OBJECT-TYPE
    SYNTAX                SEQUENCE OF BridgeStatsEntry
    ACCESS                not-accessible
    STATUS                mandatory

    DESCRIPTION
        ”A list of bridge statistics entries. The number of entries is given by the value of
        bridgeNumber. This table is only valid for the LAN ports.”
    ::=  { bridge 2 }

bridgeStatsEntry OBJECT-TYPE
    SYNTAX                    BridgeStatsEntry
    ACCESS                    not-accessible
    STATUS                    mandatory
    INDEX                         { bridgeStatsIndex }
    ::=  { bridgeStatsTable 1 }
```

```
BridgeStatsEntry ::= SEQUENCE {
    bridgeStatsIndex
        INTEGER,
    averageForwarded
        Counter,
    maxForwarded
        Counter,
    averageRejected
        Counter,
    maxRejected
        Counter,
    lanAccepts
        Counter,
    lanRejects
        Counter,
    deletedFrames
        Counter
    }

bridgeStatsIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”A unique value for each bridge statistics instance.”
    ::=  { bridgeStatsEntry 1 }
averageForwarded OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This reports the average number of frames per second accepted from the LAN for
        forwarding to the serial ports over the sampling period.”
    ::=  { bridgeStatsEntry 2 }

maxForwarded OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        ”This is the highest value of averageForwarded recorded since the previous station or
        statistics reset.”
    ::=  { bridgeStatsEntry 3 }

averageRejected OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This reports the average number of frames per second that have failed the filtering
        criterion for forwarding to the serial ports over the previous sampling period.”
    ::=  { bridgeStatsEntry 4 }

maxRejected OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter is a count of the highest value of averageRejected recorded since the
        previous station or statistics reset.”
    ::=  { bridgeStatsEntry 5 }

lanAccepts OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This variable reports the total number of frames that have been accepted from the LAN
        for forwarding over the serial links since the last station or statistics reset.”
    ::=  { bridgeStatsEntry 6 }

lanRejects OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This reports the total number of frames that have failed the filtering criterion for
        forwarding to the serial ports since the last station or statistics reset.”
    ::=  { bridgeStatsEntry 7 }

deletedLanFrames OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The number of LAN frames which have been deleted, due to an internal lack of
        resource error,  since the last station restart or the last statistics reset.”
    ::=  { bridgeStatsEntry 8 }

stpTable OBJECT-TYPE
    SYNTAX                  SEQUENCE OF StpEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        ”A list of stp entries. The number of entries is given by the value of bridgeNumber.”
    ::=  { bridge 3 }
```

```
stpEntry OBJECT-TYPE
    SYNTAX                  StpEntry
    ACCESS                  not-accessible
    STATUS                  mandatory
    INDEX                       { stpIndex }
    ::=  { stpTable 1 }

StpEntry ::= SEQUENCE {
    stpIndex
        INTEGER,
    portSpatState
        INTEGER,
    pathCostMode
        INTEGER,
    pathCostAutoValue
        INTEGER,
    pathCostManValue
        INTEGER,
    portPriorityMode
        INTEGER,
    portPriorityAutoValue
        INTEGER,
    portPriorityManualValue
        INTEGER
    }

stpIndex OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The interface on which this this entry’s equivalence is effective.”
    ::=  { stpEntry 1 }

pathCostMode OBJECT-TYPE
    SYNTAX                  INTEGER(1..2)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This variable controls the mode of determining the path cost for the port. If set to 1, the
        path cost is determined automatically by the station, if set to 2, it is configured by the
        manager.”
    ::=  { stpEntry 2 }

pathCostAutoValue OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the value of path cost assigned to the port automatically by the station.”
    ::=  { stpEntry 3 }

pathCostManualValue OBJECT-TYPE
    SYNTAX              INTEGER(0..65535)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the value of path cost assigned to the port by the manager.This parameter is also
        saved in non-volatile storage.”
    ::=  { stpEntry 4 }
portSpatState OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter indicates the spanning tree state of a particular port. It may return one of
        the following five values.

        Disabled            1

        Blocking            2

        Listening           3

        Learning            4

        Forwarding          5"

    ::=  { stpEntry 5 }

portPriorityMode OBJECT-TYPE
    SYNTAX              INTEGER(1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This variable controls the mode of determining the priority for the port. If set to 1, the
        priority is determined automatically by the station, if set to 2, it is configured by the
        manager.”
    ::=  { stpEntry 6 }

portPriorityAutoValue OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the value of priority assigned to the port automatically by the station.”
    ::=  { stpEntry 7 }

portPriorityManualValue OBJECT-TYPE
    SYNTAX              INTEGER(1..255)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the value of priority assigned to the port by the manager.”
    ::=  { stpEntry 8 }

portDesignatedRoot OBJECT-TYPE
    SYNTAX              OCTET STRING (SIZE(8))
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”The bridge identifier of the of the root bridge recorded as the rootin hellog frames
        transmitted by the bridge on this port.”
    ::=  { stpEntry 9 }

portDesignatedCost OBJECT-TYPE
    SYNTAX              INTEGER (0..65535)
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”The path cost to the root from the designated port on the lan to which this port is
        attached.”
    ::=  { stpEntry 10 }

portDesignatedPort OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”The port id of the bridge beleived to the designated port of the lan associated with the
        port.”
    ::=  { stpEntry 11 }

portDesignatedBridge OBJECT-TYPE
    SYNTAX                    OCTET STRING (SIZE(8))
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”The bridge identifier of the of the designated bridge for the lan associatedwith the port.”
    ::=  { stpEntry 12 }

spanningTree OBJECT-TYPE
    SYNTAX                    INTEGER (1..2)
    ACCESS                    read-write
    STATUS                    mandatory

    DESCRIPTION
        ”This object describes whether or not the spanning tree protocol is enabled on this
        station. A value of 1 disables the spanning tree protocol, a value of 2 enables it.”
    ::=  { bridge 4 }
```

```
spatPriority OBJECT-TYPE
    SYNTAX              INTEGER (1..65535)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”Each station has a priority, used by the spanning Tree protocol in determining the root
        bridge of a spanning tree network. The lower the value, which can range between the
        values 1 and 65535, the higher the priority.”
    ::=  { bridge 5 }

spatHelloTimer OBJECT-TYPE
    SYNTAX              INTEGER (1..10)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the period of time, in seconds, between the transmission of subsequent
        configuration BPDUs, used by the spanning tree algorithm.”
    ::=  { bridge 6 }

spatResetTimer OBJECT-TYPE
    SYNTAX              INTEGER (1..1800)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”The length of time, in seconds, after a serial link reset, for which the corresponding
        serial port’s path cost is forced to its maximum value.”
    ::=  { bridge 7 }

spatVersion OBJECT-TYPE
    SYNTAX              INTEGER {
                        revisionC(3),
                        revision8(8)
                        }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This variable determines the version of the spanning tree algorithm implemented in the
        station.”
    ::=  { bridge 8 }
```

```
spanningMcastAddr OBJECT-TYPE
    SYNTAX              OCTET STRING ( SIZE (6))
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the multicast address of the frames used by the spanning tree protocol. The
        multicast address is in hexadecimal form, in network byte order.”
    ::=  { bridge 9 }
```

```
operatingMode OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This determines whether the bridge will forward frames based only on the static
        addresses in the filtering database stored in its preconfigured table, or use its learning
        algorithm. It may take the following values:

        learning mode       1
        preconfigured mode  2"

    ::=  { bridge 10 }

preconfSourceFilter OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This determines whether the bridge will forward frames based only on the static
        addresses in the filtering database stored in its preconfigured table, or use its learning
        algorithm. This parameter determines whether the bridge will filter on destination
        address alone, or destination and source addresses. A value of 1 disables filtering on the
        source address, whilst a value of 2 enables it.”
    ::=  { bridge 11 }

typeFilter OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This parameter determines whether or not filtering is performed on frame type. A value
        of 1 disables this filtering, a value of 2 enables it.”
    ::=  { bridge 12 }

typePrioritisation OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory
```

DESCRIPTION

”This parameter enables or disables the prioritisation of traffic destined for a remote LAN, dependent upon the type field of the frame. It can take on either of the following two values.

Disabled	1
Enabled	2"

::= { bridge 13 }

```
dynamicLearningInLM OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This parameter allows the enabling and disabling of learning when the station is
        configured to be in Learning mode. A value of 1 disables learning, whilst a value of 2
        enables it.”
    ::=  { bridge 14 }

forgetAddressTimer OBJECT-TYPE
    SYNTAX              INTEGER (24..1000)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the time, in 5 second units, after which, if an address has not been seen on the
        local LAN, it is forgotten from the learning address table.”
    ::=  { bridge 15 }

deleteAddressTimer OBJECT-TYPE
    SYNTAX              INTEGER (1..20000)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the time, in 5 second units, after which, if an address has not been seen on the
        local LAN, it is deleted from the learning address table.”
    ::=  { bridge 16 }

multicastDisposition OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This parameter determines the actions of the station upon receiving a multicast frame,
        when the station is set to preconfigured mode. If it is set to 2, multicasts are forwarded
        unconditionally to the serial link. If it is set to 1, they are handled like single cast
        addresses, being forwarded only if the multicast in question is present in the filtering
        database with the appropriate disposition.”
    ::=  { bridge 17 }
```

```
filterMatches OBJECT-TYPE
    SYNTAX              INTEGER (1..2)
    ACCESS              read-write
    STATUS              mandatory
    DESCRIPTION
        ”This parameter determines the action of the station when filtering on frame type. If it is
        set to 1, the frame is discarded if the type matches, and forwarded if it does not. If it is set
        to 2, the frame is forwarded if it matches, and discarded if it does not.”
    ::=  { bridge 18 }

ieeeFormatFilter OBJECT-TYPE
    SYNTAX                  INTEGER (1..2)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This variable determines whether IEEE802.3 frames will be forwarded or discarded,
        when type filtering is being performed and a match achieved, and may assume either of
        two values:

        discard             1
        forward             2"

    ::=  { bridge 19 }

priorityMatches OBJECT-TYPE
    SYNTAX                  INTEGER (1..2)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This variable defines the priority assigned to a frame if it is being forwarded after type
        matching, and can assume either of the two following values:

        high                1
        low                 2"

    ::=  { bridge 20 }

ieeeFormatPriority OBJECT-TYPE
    SYNTAX                  INTEGER (1..2)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This variable defines the priority assigned to a IEEE 802.3 frame if it is being
```

```
          forwarded after type matching, and can assume either of the two following values:

          high                  1
          low                   2"

     ::=  { bridge 21 }

averagePeriod OBJECT-TYPE
     SYNTAX                     INTEGER
     ACCESS                     read-only
     STATUS                     mandatory

     DESCRIPTION
          ”This is the averaging period over which the statistics in the BridgeStatsTable get
          updated.”
     ::=  { bridge 22 }

triangulation OBJECT-TYPE
     SYNTAX                     INTEGER(1..2)
     ACCESS                     read-write
     STATUS                     mandatory

     DESCRIPTION
          ”This variable determines if the triangulation facility is enabled for this station, and can
          take either of the following values:

          disabled              1
          enabled               2"

     ::=  { bridge 23 }

adaptiveRouting OBJECT-TYPE
     SYNTAX                     INTEGER (1..2)
     ACCESS                     read-write
     STATUS                     mandatory

     DESCRIPTION
          ”This variable determines whether the adaptive routing protocol is enabled for this
          station, and can assume either of the values:


          disabled              1
```

```
        enabled              2

        It should be noted that in order to enable adaptive routing, the spanning tree algorithm
        must  also be enabled.”
    ::=  { bridge 24 }

adaptiveMcastAddr OBJECT-TYPE
    SYNTAX                OCTET STRING ( SIZE (6))
    ACCESS                read-write
    STATUS                mandatory

    DESCRIPTION
        ”This is the multicast address of the frames used by the adaptive routing protocol. The
        multicast address is in hexadecimal form, in network byte order.”
    ::=  { bridge 25 }

arAddressInfo OBJECT IDENTIFIER ::= { bridge 26 }

standbyRemote OBJECT-TYPE
    SYNTAX                INTEGER
    ACCESS                read-only
    STATUS                mandatory

    DESCRIPTION
        ”This object reports the total number of Adaptive Routing Destination Routes.”
    ::=  { arAddressInfo 1 }
```

```
standbyLocal OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This object reports the total number of Adaptive Routing Source Routes.”
    ::=  { arAddressInfo 2 }

activeRemote OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This object reports the total number of Adaptive Routing Remote Addresses.”
    ::=  { arAddressInfo 3 }

activeLocal OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This object reports the total number of Adaptive Routing Learned Addresses.”
    ::=  { arAddressInfo 4 }

maxSerialLoading OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object is used by the Adaptive Routing protocol. If the loading of a serial link
        exceeds this value, then any frames enqueued for this link by Adaptive Routing will be
        removed from the queue and enqueued on the other serial link. The value of this object is
        a measure of the percentage loading of the line.”
    ::=  { bridge 27 }

serialLoadPeriod OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the sampling period over which the serial links loading is measured, in seconds.”
    ::=  { bridge 28 }

serialLoading OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”Each instance of this object gives the percentage loading on a particular serial line. The
        instances are indexed by the value of if Index.”
    ::=  { bridge 29 }

SYNTAX SEQUENCE OF SerialLoadValue
    ACCESS              not-accessible
    STATUS              mandatory
    INDEX                   { phySerIfEntry  }

    DESCRIPTION
        ”List of serial load statistics.”
    ::=  { bridge 29 }

SerialLoadValue ::= SEQUENCE {

serialLoadTable INTEGER }

serialLoading OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”Each instance of this object gives the percentage loading on a particular serial line. The
        instances are indexed by the value of if Index.”
    ::=  { serialLoadTable 1 }

filteringDataBaseTable OBJECT IDENTIFIER ::= { bridge 30 }

filteringDbTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF FilteringDbEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        "A list of filtering  data base entries."
    ::=  { filteringDataBaseTable 1 }

filteringDbEntry OBJECT-TYPE
    SYNTAX                  SEQUENCE OF FilteringDbEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        "This OCTET STRING contains one entry from the table. Each entry consists of a MAC
        address, filter disposition, filter status and type."
    INDEX                       { filteringDbMacAddress }
    ::=  { filteringDbTable 1 }
```

```
FilteringDbEntry ::= SEQUENCE {
    filteringDbMacAddress
        OCTET STRING,
    filteringDbDisposition
        INTEGER,
    filteringDbStatus
        INTEGER,
    filteringDbType
        INTEGER
    }

filteringDbMacAddress OBJECT-TYPE
    SYNTAX              OCTET STRING(6)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This OCTET STRING contains the MAC address in hexadecimal form, in network byte
        order.”
    ::=  { filteringDbEntry 1 }

filteringDbDisposition OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”The Filter Disposition byte has the following values :

        0   discard
        1   lan
        2   channel1
        4   channel2
        7   flood”

    ::=  { filteringDbEntry 2 }

filteringDbStatus OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
```

”The following values are available :

1 Learnt

2 Unlearnt

3 Static

4 Bridge Multicast address

5 Learnt - ar controlled

6 Unlearnt - ar controlled

7 Static - ar controlled

```
        Only static entries may be entered in the table, remotely, via SNMP., i.e. a SET on this
        object may only have a value of 3.”
    ::=  { filteringDbEntry 3 }

filteringDbType OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”The type field can take either of the values shown below:


        valid    1

        invalid 2


        For a Get or GetNext, this field is always returned as valid. For a Set, if the field is set to
        a value of valid, an entry is added to the table, whilst if it is set to invalid an entry is
        deleted.”
    ::=  { filteringDbEntry 4 }

filteringDbAction OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This object is used to clear the Filtering Database Table of all static entries. Other
        entries are not affected.Only two values are valid to use with this object. A value of 1 is
        always returned when a GET is performed upon the object. To clear all static entries
        from the table, a value of 2 must be used in the SET.  A SET with a value of 1 has no
        effect.”
    ::=  {filteringDataBaseTable 2 }

filteringDbPage OBJECT-TYPE
    SYNTAX                  SEQUENCE OF FilteringDbPageValue
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
```

```
        ”A list of paged entries from the filtering data base table. We shoulduse an object
        ‘filteringDbPageIndex’ as the index if one such exists, but since it doesnt we cannot.”
    ::=  { filteringDataBaseTable 3 }

FilteringDbPageValue ::= SEQUENCE {
    filteringDbPage OCTET STRING ( SIZE (0..640))
    }

filteringDbPageValue OBJECT-TYPE
    SYNTAX                  OCTET STRING (SIZE(0..640))
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This OCTET STRING contains up to 80 valid table entries. Each entry consists of a
        MAC address, filter disposition and filter status. Shown below is the first valid entry of a
        get on this object, as it is returned within the OCTET STRING :


        Bytes 0 - 5         Byte 6                  Byte 7

        MAC Address         Filter Disposition      Filter Status


        Each entry in the OCTET STRING is in the same order as that shown for the first entry
        above. The MAC address is in hexadecimal form, in network byte order.The Filter
        Disposition byte has the following values :


        0       discard

        1       lan

        2       channel1

        4       channel2

        7       flood


        The filter status byte can take the following values:
```

```
        1       Learnt

        2       Unlearnt

        3       Static

        4       Bridge Multicast address

        5       Learnt - ar controlled

        6       Unlearnt - ar controlled

        7       Static - ar controlled”

    ::=  { filteringDbPage 1 }

priorityTable  OBJECT IDENTIFIER ::= { bridge 31 }

priorityTableEntry OBJECT-TYPE
    SYNTAX                  PriorityTableEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        ”This sequence defines one entry in the priority table.There is a maximum of 8 instances
        of this sequence.”
    INDEX                       { priorityTableEntryValue }
    ::=  { prioritySubTable 1 }

PriorityTableEntry ::= SEQUENCE {
    priorityTableEntryValue
        INTEGER,
    priorityTableEntryType
        INTEGER
    }
```

```
priorityTableEntryValue OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”The priority being added to, deleted or read from the table.”
    ::=  { priorityTableEntry 1 }

priorityTableEntryType OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object defines whether a priority is being added to or deleted from the priority
        table. It can assume the following values:

        valid               1
        invalid             2


        A value of valid causes a priority to be added to the table, while a value of invalid causes
        it to be deleted. For a Get or Getnext, the value returned is always valid.”
    ::=  { priorityTableEntry 2 }

priorityTableAction OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object is used to clear the priority Table. Only two values are valid to use with this
        object. A value of 1 is always returned when a GET is performed upon the object. To
        clear the table,  a value of 2 must be used in the SET.  A SET with a value of 1 has no
        effect.”
    ::=  { priorityTable 2 }

filterTable OBJECT IDENTIFIER ::= { bridge 32 }

filterSubTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF FilterTableEntry
```

```
ACCESS                  not-accessible
STATUS                  mandatory

DESCRIPTION
   ”This is a list of entries in the filter table.”
::=  { filterTable 1 }
```

```
filterTableEntry OBJECT-TYPE
    SYNTAX                      SEQUENCE OF FilterTableEntry
    ACCESS                      not-accessible
    STATUS                      mandatory
    INDEX                           { filterTableEntryValue }
    ::=  { filterSubTable 1 }

FilterTableEntry ::= SEQUENCE {
    filterTableEntryValue
        INTEGER,
    filterTableEntryType
        INTEGER
    }

filterTableEntryValue OBJECT-TYPE
    SYNTAX                      INTEGER
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”This sequence defines one entry in the filter table.There is a maximum of 8 instances of
        this object.”
    ::=  { filterTableEntry 1 }

filterTableEntryType OBJECT-TYPE
    SYNTAX                      INTEGER
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”This determines whether the value of a filter to be added to,  or deleted from. It can
        assume the following values:

        valid                   1
        invalid                 2


        A value of valid causes the filter to be added. A value of invalid causes it to be deleted.
        For a Get, this value is always returned as valid.”
    ::=  { filterTableEntry 2 }

filterTableAction OBJECT-TYPE
```

```
SYNTAX                INTEGER
ACCESS                read-write
STATUS                mandatory

DESCRIPTION
    ”This object is used to clear the filter Table of its static entries, all others are not
    affected.Only two values are valid to use with this object. A value of 1 is always returned
    when a GET is performed upon the object. To clear the table of static entries, a value of
    2 must be used in the SET.  A SET with a value of 1 has no effect.”
::=  { filterTable 2 }
```

```
brdgBroadcastRX OBJECT-TYPE
    SYNTAX                    INTEGER (1..2)
    ACCESS                    read-write
    STATUS                    mandatory

    DESCRIPTION
        ”When set to 1 indicates that broadcast packets will be received onthe lan channel, when
        set to 2 they will be ignored.”
    ::=  { bridge 33 }

brdgLiveSpatHelloTimer OBJECT-TYPE
    SYNTAX                    INTEGER (1..10)
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”The time, in seconds, between the transmission of configuration BPDU’sused by
        spanning tree.  If this bridge is the root bridge then all bridgeswill use this value.”
    ::=  { bridge 34 }

brdgHoldTime OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”After this time, in seconds, unacknowledged spanning tree bpdus willbe reset.”
    ::=  { bridge 35 }

brdgMaxAge OBJECT-TYPE
    SYNTAX                    INTEGER (6..40)
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”The max time, in seconds, that a received STP protocol message can exist before it is
        discarded by the bridge.  If this bridge is theroot bridge, then this value will be used by
        all bridges in the network.”
    ::=  { bridge 36 }

brdgLiveMaxAge OBJECT-TYPE
    SYNTAX                    INTEGER (6..40)
```

```
ACCESS              read-only
STATUS              mandatory

DESCRIPTION
    ”The max time, in seconds, that a received STP protocol message can exist before it is
    discarded by the bridge.  If this bridge is theroot bridge, then this value will be used by
    all bridges in the network. This is the current operation value and will be the sameas the
    initial Max Age of the root bridge of the network.”
::=  { bridge 37 }
```

```
brdgLiveBridgeForwardDelay OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”The time spent in seconds in STP listening and learning states bythe bridge.  Thisis the
        current operation value and is assigned to thisbridge by the root bridge if this bridge is
        not the root.”
    ::=  { bridge 38 }

brdgTopologyChangeCount OBJECT-TYPE
    SYNTAX              COUNTER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”The count of the number of STP network topology changes since the lastbridge reset.”
    ::=  { bridge 39 }

brdgPriority OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”The spanning tree priority of the bridge.”
    ::=  { bridge 40 }

brdgPathCostScalingFactor OBJECT-TYPE
    SYNTAX              INTEGER (1..10000)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”A scaling factor to be used when calculating path costs.  The IEEE specification (Draft
        9) recommends a method for calculating the path cost for spanning tree ports.  This
        caculation is:

        path cost = 1000 / data rateLan speed in Megabits/sec


        It is the ‘1000’ of the above calculation which will be replaced by the configured value
        of brdgPathCostScalingFactorthe value of thisobject will be used for.”
    ::=  { bridge 40 }
```

Boot group

```
bootpRetryCount  OBJECT-TYPE
    SYNTAX                  INTEGER ( 0..65535 )
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”The number of times a Bootp Request will be transmitted before the Bootp attempt is
        considered to have failed.  If this parameter is set to 0, Bootp is skipped and an
        immediate TFTP download attempted.”
    ::= { boot 1 }

downloadRetryCount  OBJECT-TYPE
    SYNTAX                  INTEGER ( 0..65535 )
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”The number of times the complete Bootp/TFTP download cycle is attempted before the
        download attempt is considered to have failed.”
    ::= { boot 2 }

downloadFilename  OBJECT-TYPE
    SYNTAX                  OCTET STRING (SIZE( 0..128))
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This character string is inserted in the ‘boot file name’ field of the Bootp Request; its
        semantics is determined by the Boot Server.  If Bootp is skipped and an immediate TFTP
        download attempted, this parameter is used by TFTP as the name of the file to be
        downloaded.”
    ::= { boot 3 }

bootserverIpAddress  OBJECT-TYPE
    SYNTAX                  IpAddress
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”If this address is null (0.0.0.0), the Boot Request is sent to the IP broadcast address as
        determined by ipAdEntBcastAddr.  Otherwise it is sent to this address.  It is known that
        certain bootserver implementations do not respond to the standard limited broadcast,
        hence the need for this parameter.”
    ::= { boot 4 }

loadserverIpAddress  OBJECT-TYPE
    SYNTAX                  IpAddress
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”If this address is null (0.0.0.0), TFTP will request a download from the server address
        returned in the Bootp Response.  Otherwise it will use this address.”
    ::= { boot 5 }

uniqueBroadcastAddress  OBJECT-TYPE
    SYNTAX                  IpAddress
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”A unique broadcast address specified by the user which does not conform to the
        standard ip broadcast address format. If the user specifies a unique broadcast address, this
        will take precedence over the standard ip broadcast address.”
    ::= { boot 6 }

tftpRetryCount  OBJECT-TYPE
    SYNTAX                  INTEGER ( 0..65535)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”The number of times a PDU will be retransmitted during a TFTP transfer before
        declaring the transfer to have failed.”
    ::= { boot 7 }

tftpRetryPeriod  OBJECT-TYPE
    SYNTAX                  INTEGER (0..65535)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”The time  in units of 50ms that TFTP will wait for the next requested block before
        initiating error recovery.”
    ::= { boot 8 }

genericFilename  OBJECT-TYPE
    SYNTAX                  OCTET STRING (SIZE( 0..128))
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”The bootfilename returned from the boot server following a bootprequest which will be
        used in a subsequent tftp download request.”
    ::= { boot 9 }
```

RmtLapb grouptc "RmtLapb group"§

Remote lapb configuration tabletc "Remote lapb configuration table"§

```
rmtLapbConfigTable OBJECT-TYPE
    SYNTAX                  SEQUENCE OF RmtLapbCTEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        ”A list of configurable LapbCT entries, one entry per interface.”
    ::= { rmtLapb 1 }

rmtLapbCTEntry OBJECT-TYPE
    SYNTAX                  RmtLapbCTEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        ”A list of configurable data, for each lapb interface.”
    INDEX                       { rmtLapbCTIndex }
    ::= { rmtLapbConfigTable 1 }
```

```
RmtLapbCTEntry ::= SEQUENCE {
    rmtLapbCTIndex
        INTEGER,
    rmtLapbCTLinkAddr
        INTEGER,
    rmtLapbCTExtSeqNumbering
        INTEGER,
    rmtLapbCTWindow
        INTEGER( 1..127 ),
    rmtLapbCTModeT1
        INTEGER,
    rmtLapbCTManualT1Value
        INTEGER( 10..5000 ),
    rmtLapbCTT3LinkIdleTimer
        INTEGER( 1..65535 ),
    rmtLapbCTN2RetryCount
        INTEGER( 1..255 ),
    rmtLapbCTLinkReset
        INTEGER,
    rmtLapbCTX25PortLineSpeed
        INTEGER,
    rmtLapbCTInitLinkConnect
        INTEGER
    }

rmtLapbCTIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This defines the interface for which this entry is valid.”
    ::= { rmtLapbCTEntry 1 }

rmtLapbCTLinkAddr OBJECT-TYPE
    SYNTAX              INTEGER ( 1..2 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object defines the link level logical address, and also the packet level logical
        address. It can take the following values :-

        DTE                     1
        DCE                     2"

    ::= { rmtLapbCTEntry 2 }

rmtLapbCTExtSeqNumbering OBJECT-TYPE
    SYNTAX                      INTEGER ( 1..2 )
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”This object enables/disables the use of extended frame sequence numbering at the link
        level. When enabled, modulo 128 numbering is used, as opposed to modulo 8 when
        disabled. It can take the following values :-

        disabledenabled       1
        enableddisabled       2"

    ::= { rmtLapbCTEntry 3 }

rmtLapbCTWindow OBJECT-TYPE
    SYNTAX                      INTEGER (1..127)
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”This parameter represents the maximum number of I-frames which can be transmitted
        before an acknowledgement is received.  It can have any value between 1 and 127, ( for
        this value to be greater than 7, extended sequence numbering must be invoked. This is
        the value of k as defined in the 1984 CCITT Recommendation X.25 VOLUME VIII -
        FASCILE VIII.3"
    ::= { rmtLapbCTEntry 4 }

rmtLapbCTModeT1 OBJECT-TYPE
    SYNTAX                      INTEGER (1..2)
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”If the  value of modeT1 is 1 then T1 timer is determined automatically by the software.
        If the value of this object is 2 then the timer is settable by the manager. The value of T1
        won’t change until the next reset of the station or the next link reset.”
    ::= { rmtLapbCTEntry 5 }

rmtLapbCTManualT1Value OBJECT-TYPE
    SYNTAX                  INTEGER (10..5000)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This object allows the manager to set the value of the T1 timer as defined in the 1984
        CCITT Recommendation X.25 VOLUME VIII -  FASCICLE VIII.3. The settable range
        is 10 - 5000 which represents the timer value in milliseconds. This object maybe set at
        any time, but it is only used if the value of modeT1 is 1.  This value will not become
        active until the next link or station reset.”
    ::= { rmtLapbCTEntry 6 }

rmtLapbCTT3LinkIdleTimer OBJECT-TYPE
    SYNTAX                  INTEGER (1..65535)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This is the value of the T3 timer, which defines the maximum time which a link can
        remain in the idle condition before being identified as non-operational.”
    ::= { rmtLapbCTEntry 7 }

rmtLapbCTN2RetryCount OBJECT-TYPE
    SYNTAX                  INTEGER (1..255)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This is the value of N2 as defined in the 1984 CCITT Recommendation X.25 VOLUME
        VIII -  FASCICLE VIII.3"
    ::= { rmtLapbCTEntry 8 }

rmtLapbCTLinkReset OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
```

```
        ”This parameter causes a link reset. The link reset occurs after a pre-determined period of
        time which is implementation specific.A value of 2 is used to SET this object and to
        cause the link reset. A value of 1 will be returned when a GET is performed on this
        object. If SET with a value of 1, no action is taken. No other values are valid.”
    ::= { rmtLapbCTEntry 9 }

rmtLapbCTX25PortLineSpeedOBJECT-TYPE
    SYNTAX                      INTEGER  {
        b1200 (1200),
        b2400 (2400),
        b4800 (4800),
        b9600 (9600),
        b19200 (19200),
        b32000 (32000),
        b48000 (48000),
        b64000 (64000)
        }
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”The configured line speed.”
    ::= { rmtLapbCTEntry 10 }

rmtLapbCTInitLinkConnectOBJECT-TYPE
    SYNTAX                      INTEGER  {
        enabled (1),
        disabled (2)
        }
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        ”Instructs the serial code to initiate a link connect, as opposed to waiting to be connected
        to. Effective only when x25 is enabled.”
    ::= { rmtLapbCTEntry 11 }

remote lapb statistics tabletc "remote lapb statistics table"§

rmtLapbStatsTable OBJECT-TYPE
    SYNTAX                      SEQUENCE OF RmtLapbSTEntry
    ACCESS                      not-accessible
    STATUS                      mandatory
    DESCRIPTION
        ”A list of LAPB interface entries.”
    ::= { rmtLapb 2 }

rmtLapbSTEntry OBJECT-TYPE
    SYNTAX              RmtLapbSTEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”The LAPB table. An entry in this table is uniquely identified by the value of the
        rmtLapbSTIndex variable associated with the interface to which the statistics refer.”
    INDEX                   { rmtLapbSTIndex }
    ::= { rmtLapbStatsTable 1 }

RmtLapbSTEntry ::= SEQUENCE {
    rmtLapbSTIndex
        INTEGER,
    rmtLapbSTState
        INTEGER,
    rmtLapbSTAutoT1value
        INTEGER,
    rmtLapbSTLastResetTime
        OCTET STRING (SIZE (13)),
    rmtLapbSTLastResetReason
        INTEGER,
    rmtLapbSTResetCount
        Counter,
    rmtLapbSTSendFrameCount
        Counter,
    rmtLapbSTRcvFrameCount
        Counter,
    rmtLapbSTSendOctetCount
        Counter,
    rmtLapbSTRcvOctetCount
        Counter,
    rmtLapbSTAbortCount
        Counter,
    rmtLapbSTCrcErrorCount
        Counter
    }
```

```
rmtLapbSTIndex OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The interface on which this entry’s equivalence is effective. This object identifies the
        same physical interface as identified by the phySerIfIndex.”
    ::=  { rmtLapbSTEntry 1 }

rmtLapbSTState OBJECT-TYPE
    SYNTAX                  INTEGER (1..4)
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This parameter indicates the state of the LAPB link, it can take on one of the following
        four values.


        inactive                1

        active  2

        unused 3

        tooFast4


        Inactive indicates that LAPB initialisation was unsuccessful. Active indicates that LAPB
        initialisation on this link was successful. Unused indicates that, in the case where there
        are two parallel links between bridges one of which is slower, the slower link has been
        tagged ‘unused’. This is done to concentrate transmissions on the faster link and thus
        avoid a heavy re-ordering overhead due to frames arriving out of sequence. TooFast
        indicates that the clock on the received signal is too fast for the interface in question.”
    ::= { rmtLapbSTEntry 2 }

rmtLapbSTAutoT1value OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This parameter is the value of the T1 timer as defined in the 1984 CCITT
        Recommendation X.25 VOLUME VIII -  FASCICLE VIII.3 and is determined
        empirically by the software. It is the actual value of the timer in milliseconds.”
    ::= { rmtLapbSTEntry 3 }

rmtLapbSTLastResetTime OBJECT-TYPE
    SYNTAX                OCTET STRING (SIZE (13))
    ACCESS                read-only
    STATUS                mandatory

    DESCRIPTION
        ”Time and date when the last link reset occurred. The format of this string is the same as
        that defined for stationTime.”
    ::= { rmtLapbSTEntry 4 }

rmtLapbSTLastResetReason OBJECT-TYPE
    SYNTAX                INTEGER (1..9)
    ACCESS                read-only
    STATUS                mandatory

    DESCRIPTION
        ”The reason for the link reset. This number can have any one of nine values. The
        following is a list of the values and reset reasons they indicate :

        noReason                 1
        xpcError                 2
        ctsLost                  3
        retryLimitExceeded       4
        receivedIdle             5
        personalityModule        6
        softwareWatchdog         7
        nmcInitiated             8
        invalidConfiguration     9"

    ::= { rmtLapbSTEntry 5 }

rmtLapbSTResetCount OBJECT-TYPE
    SYNTAX                Counter
    ACCESS                read-only
    STATUS                mandatory

    DESCRIPTION
        ”The parameter represents the number of link resets performed since the last warm or
        cold restart of the station or the last statistics reset.”
    ::= { rmtLapbSTEntry 6 }

rmtLapbSTSendFrameCount OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The parameter represents the number of I-frames sent since the last station restart or the
        last statistics reset.”
    ::= { rmtLapbSTEntry 7 }

rmtLapbSTRcvFrameCount OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The parameter represents the number of I-frames sent since the last station restart or the
        last statistics reset.”
    ::= { rmtLapbSTEntry 8 }

rmtLapbSTSendOctetCount OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The parameter represents the number of bytes sent since the last station restart  or the
        last statistics reset.”
    ::= { rmtLapbSTEntry 9 }

rmtLapbSTRcvOctetCount OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The parameter represents the number of bytes received since the last station restart or
        the last statistics reset.”
    ::= { rmtLapbSTEntry 10 }
```

```
rmtLapbSTAbortCount OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This parameter represents the number of aborts received since last station restart or the
        last statistics reset.”
    ::= { rmtLapbSTEntry 11 }

rmtLapbSTCrcErrorCount OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”The parameter represents the number of CRC errors received since the last station
        restart or the last statistics reset.”
    ::= { rmtLapbSTEntry 12 }
```

X25 Grouptc "X25 Group"§

```
x25Operation OBJECT-TYPE
    SYNTAX              INTEGER ( 1..2 )
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This object tells us whether X25 is enabled or disabled, at the packet level, for the
        device. The following values apply :-

        enabled             1,
        disabled            2


        If X25PLP is turned on, this markedly affects the manner in which Lapb works on this
        interface. It also affects the relevance and effectiveness of some Lapb managed objects.”
    ::= { x25 1 }

x25OperNextReset OBJECT-TYPE
    SYNTAX              INTEGER ( 1..2 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object tells us whether X25 is to be enabled or disabled, at the packet level,  for the
        device on the next Reset. The following values apply :-

        enabled             1,
        disabled            2


        If X25PLP is turned on, this markedly affects the manner in which Lapb works on this
        interface. It also affects the relevance and effectiveness of some Lapb managed objects.”
    ::= { x25 2 }
```

x25 Connection set-up table

```
x25ConSetupTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF  X25CSTEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”This table holds configuration data used in setting up each X25 SVC.”
    ::= { x25 3 }
```

```
x25CSTEntry OBJECT-TYPE
    SYNTAX              X25CSTEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”Each entry in the Table includes all the configurable objects for each interface.”
    INDEX                   { x25CSTIndex }
    ::= { x25ConSetupTable 1 }

X25CSTEntry ::= SEQUENCE {
    x25CSTIndex
        INTEGER,
    x25CST8084Switch
        INTEGER,
    x25CSTSrcDTEAddr
        OCTET STRING ( SIZE (15) ),
    x25CSTDestDTEAddr
        OCTET STRING ( SIZE (15) ),
    x25CST2WayLgclChanNum
        INTEGER,
    x25CSTPktSeqNumFlg
        INTEGER,
    x25CSTFlowCntrlNeg
        INTEGER,
    x25CSTDefaultWinSize
        INTEGER,
    x25CSTDefaultPktSize
        INTEGER,
    x25CSTNegWinSize
        INTEGER,
    x25CSTNegPktSize
        INTEGER,
    x25CSTCUGSub
        INTEGER,
    x25CSTLclCUGValue
        INTEGER,
    x25CSTRvrsChrgReq
        INTEGER,
    x25CSTRvrsChrgAcc
        INTEGER
    }
```

```
x25CSTIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory
    DESCRIPTION
        ”This index defines the interface for which this entry is valid. This index is  equivalent to
        the physSerIfNumber index representing the physical serial  interfaces on the device.”
    ::= { x25CSTEntry 1 }

x25CST8084Switch OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This object is an operation switch, allowing the user to change the operating mode of
        x25 on this interface. A value of 1 sets the operating mode to 1980 operating mode,
        wheras a value of 2 sets it to 1984. “
    ::= { x25CSTEntry 2 }

x25CSTSrcDTEAddr OBJECT-TYPE
    SYNTAX                  OCTET STRING
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This is the address of the Source DTE associated with this interface. It can be any
        number, up to 15 ascii characters long, in the range 0-9.”
    ::= { x25CSTEntry 3 }

x25CSTDestDTEAddr OBJECT-TYPE
    SYNTAX                  OCTET STRING
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This is the address of the Destination DTE associated with this interface. It can  be any
        number, up to 15 ascii characters long, in the range 0-9.”
    ::= { x25CSTEntry 4 }

x25CST2WayLgclChanNum OBJECT-TYPE
    SYNTAX                  INTEGER ( 1.. 4095 )
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
```

```
        ”This number uniquely identifies the two-way logical channel number to be used in the
        DTE/DCE interface when establishing an SVC on this interface.”
    ::= { x25CSTEntry 5 }

x25CSTPktSeqNumFlg OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This flag allows the user to switch between Basic( modulo 8 ) and Extended ( modulo
        128 ) Packet Sequence Numbering. A value of 1 indicates the use of Basic numbering,
        while a value of 4 specifies Extended numbering.”
    ::= { x25CSTEntry 6 }

x25CSTFlowCntrlNeg OBJECT-TYPE
    SYNTAX              INTEGER ( 1..2 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object enables/disables Flow Control Negotiation on this interface. It can  have the
        following values:-

        enabled             1
        disabled            2


        If disabled, the values of window size and packet size used on the connection are the
        specifed default values. If Negotiation is enabled, then this DTE offers its negotiation
        values and these may be negotiated downwards by the remote entity.”
    ::= { x25CSTEntry 7 }

x25CSTDefaultWinSize OBJECT-TYPE
    SYNTAX              INTEGER ( 1..127 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the window size used if Negotiation is not enabled. If basic frame numbering is
        in operation, then it cannot have a value greater than 7. If extended frame numbering is
        enabled, then it can have a value up to  127.”
    ::= { x25CSTEntry 8 }
```

```
x25CSTDefaultPktSize OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the packet size used if Negotiation is not enabled. This must be less then or equal
        to the Physical Block size on the device. It can take the following values :

        1 - 128 bytes.
        2- 256 bytes.
        3 - 512 byrtes.”

    ::= { x25CSTEntry 9 }

x25CSTNegWinSize OBJECT-TYPE
    SYNTAX              INTEGER ( 1..127 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the window size requested when a connection is being initialised, if  Negotiation
        is enabled. If basic frame numbering is in operation, then it cannot have a value greater
        than  7. If extended frame numbering is enabled, then it can have a value up to 127.”
    ::= { x25CSTEntry 10 }

x25CSTNegPktSize OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is the packet size requested at call setup time, if Negotiation is enabled. This
        should not be larger than the Physical Block size defined for this device. It can take the
        following values:

        1 - 128 bytes.
        2 - 256 bytes.
        3 - 512 bytes.”

    ::= { x25CSTEntry 11 }
```

```
x25CSTCUGSub OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object defines whether we enable the use of the Closed User Group  selectrion
        facility for connections on this interface. It can have the following  values :-

        disabled            1
        CUGonly             4"

    ::= { x25CSTEntry 12 }

x25CSTLclCUGValue OBJECT-TYPE
    SYNTAX              INTEGER ( 0..99 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This value is the number which uniquely defines the Closed User Group value to be
        used within the Closed User Group Selection facility of the call packet.”
    ::= { x25CSTEntry 13 }

x25CSTRvrsChrgReq OBJECT-TYPE
    SYNTAX              INTEGER ( 1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”When a connection is being set up, this device will request that Reverse  Charging of the
        call be accepted by the remote end, if this object is enabled.  It can have the following
        values:-

        enabled             1
        disabled            2"

    ::= { x25CSTEntry 14 }

x25CSTRvrsChrgAcc OBJECT-TYPE
    SYNTAX              INTEGER ( 1..2)
    ACCESS              read-write
    STATUS                    mandatory

    DESCRIPTION
        ”When a connection is being set up, this box will accept Reverse Charging of the  call,
        initiated by the remote end, if this object is enabled.  It can have the following values:-

        enabled               1
        disabled              2"

    ::= { x25CSTEntry 15 }
```

x25 Connection Control Table

```
x25ConControlTable OBJECT-TYPE
    SYNTAX                    SEQUENCE OF X25CCTEntry
    ACCESS                    not-accessible
    STATUS                    mandatory

    DESCRIPTION
        ”This table holds configurable data used in controlling each X25 SVC.”
    ::= { x25 4 }

x25CCTEntry OBJECT-TYPE
    SYNTAX                    X25CCTEntry
    ACCESS                    not-accessible
    STATUS                    mandatory

    DESCRIPTION
        ”Each entry contains configurable objects allowing the control of the status of the  SVC.”
    INDEX                         { x25CCTIndex }
    ::= { x25ConControlTable 1 }

X25CCTEntry ::= SEQUENCE {
    x25CCTIndex
        INTEGER,
    x25CCTManualConnect
        INTEGER,
    x25CCTManualDisconnect
        INTEGER,
    x25CCTCfgAutoConRetry
        INTEGER,
    x25CCTOperAutoConRetryFlg
        INTEGER,
    x25CCTAutoConRetryTimer
        INTEGER,
    x25CCTMaxAutoConRetries
        INTEGER,
    x25CCTCfgTODControl
        INTEGER,
    x25CCTAutoTODControlFlg
        INTEGER,
    x25CCTTODToConnect
        OCTET STRING ( SIZE(4) ),
    x25CCTTODToDisconnect
        OCTET STRING ( SIZE(4) )
    }

x25CCTIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This index defines the interface for which this entry is valid. This index is  equivalent to
        the phySerIfNumber index representing the physical serial  interfaces on the device.”
    ::= { x25CCTEntry 1 }

x25CCTManualConnect OBJECT-TYPE
    SYNTAX              INTEGER ( 1..2)
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object, when set, causes a manual (re)connect of the X25 SVC .  The object  can
        have the following values:-

        no action           1
        connect             2
        A Get on this object will return the value 1, i.e. no action, A Set with a value of 2,  will
        cause a (re)connect of the SVC. A Set with a value of 1 shall have no effect.”
    ::= { x25CCTEntry 2 }
```

```
x25CCTManualDisconnect OBJECT-TYPE
    SYNTAX                  INTEGER ( 1..2)
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This object, when set, causes a manual disconnect of the X25 SVC .  The object  can
        have the following values:-

        no action           1
        connect             2


        A Get on this object will return the value 1, i.e. no action, A Set with a value of 2,  will
        cause a disconnect of the SVC. A Set with a value of 1 shall have no effect.”
    ::= { x25CCTEntry 3 }

x25CCTCfgAutoConRetry OBJECT-TYPE
    SYNTAX                  INTEGER ( 1..2 )
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This object will allow retries to take place when an SVC is being established. The object
        can have the following values:-

        disabled            1,
        enabled             2 “

    ::= { x25CCTEntry 4 }

x25CCTOperAutoConRetryFlgOBJECT-TYPE
    SYNTAX                  INTEGER ( 1..2 )
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”The current operation state of the x25 connection retry flag . The object can have the
        following values:-

        disabled            1,
        enabled             2 “
    ::= { x25CCTEntry 5 }

x25CCTAutoConRetryTimer OBJECT-TYPE
    SYNTAX                  INTEGER ( 1..3600 )
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This is the retry timer, which defines the delay, in seconds, between  consecutive
        attempts to establish the X25 SVC.”
    ::= { x25CCTEntry 6 }
```

```
x25CCTMaxAutoConRetries  OBJECT-TYPE
    SYNTAX                INTEGER ( 0..255 )
    ACCESS                read-write
    STATUS                mandatory

    DESCRIPTION
        ”This defines the maximum no of attempts to set up a single X25 SVC allowed. A  value
        of 0 means there is no limit to the No. of retries allowed.”
    ::= { x25CCTEntry 7 }

x25CCTCfgTODControl OBJECT-TYPE
    SYNTAX                INTEGER ( 1..2 )
    ACCESS                read-write
    STATUS                mandatory

    DESCRIPTION
        ”This object enables/disables the Time-of-Day Control  mechanism, which can  control
        the time period for which an SVC may be active, between two devices. It  can have the
        following values:-

        disabled          1
        enabled           2"

    ::= { x25CCTEntry 8 }

x25CCTAutoTODControlFlg OBJECT-TYPE
    SYNTAX                INTEGER ( 1..2 )
    ACCESS                read-write
    STATUS                mandatory

    DESCRIPTION
        ”The current operation state of the time of day control flag. This is the current
        operational value. It  can have the following values:-

        disabled          1
        enabled           2"

    ::= { x25CCTEntry  9 }

x25CCTTODToConnect OBJECT-TYPE
    SYNTAX                OCTET STRING ( 5 )
    ACCESS                read-write
    STATUS                mandatory
```

```
DESCRIPTION
     ”This defines the time of day at which the Time-of-Day Control mechanism, when
     enabled, will attempt to establish an SVC to the remote end. If the SVC  exists at this
     time, then no action is taken. The format of this time should be as  follows :-

     hh:mm


     and should take values between 00:00 and 23:59 inclusive. These four octets  should be
     ascii characters within the ranges above.”
::= { x25CCTEntry 10 }
```

```
x25CCTTODToDisconnect  OBJECT-TYPE
    SYNTAX                    OCTET STRING ( 5 )
    ACCESS                    read-write
    STATUS                    mandatory

    DESCRIPTION
        ”This defines the time of day at which the Time-of-Day Control mechanism, when
        enabled, will attempt to disconnect an SVC to the remote end. If the SVC is  already
        disconnected at this time, then no action is taken. The format of this time  should be as
        follows :-

        hh:mm


        and should take values between 00:00 and 23:59 inclusive.These four octets  should be
        ascii characters within the ranges above.”
    ::= { x25CCTEntry 11 }
```

x25 Timer Table

```
x25TimerTable  OBJECT-TYPE
    SYNTAX                    SEQUENCE OF X25TTEntry
    ACCESS                    not-accessible
    STATUS                    mandatory

    DESCRIPTION
        ”This table holds the configurable timers used with each X25 connection  (SVC).”
    ::= { x25 5 }

x25TTEntry OBJECT-TYPE
    SYNTAX                    X25TTEntry
    ACCESS                    not-accessible
    STATUS                    mandatory

    DESCRIPTION
        ”Each entry holds the configurable values of timers and retry limits in operation  for each
        SVC.”
    INDEX                         { x25TTIndex }
    ::= { x25TimerTable 1 }

X25TTEntry ::= SEQUENCE {
    x25TTIndex
        INTEGER,
    x25TTT20Timer
        INTEGER,
    x25TTT21Timer
        INTEGER,
    x25TTT22Timer
        INTEGER,
    x25TTT23Timer
        INTEGER,
    x25TTR20Limit
        INTEGER,
    x25TTR22Limit
        INTEGER,
    x25TTR23Limit
        INTEGER
    }

x25TTIndex  OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory
    DESCRIPTION “”
    ::= { x25TTEntry 1 }

x25TTT20Timer  OBJECT-TYPE
    SYNTAX                  INTEGER ( 1..1024 )
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This defines the value of the restart retransmit timer, in seconds.”
    ::= { x25TTEntry 2 }

x25TTT21Timer  OBJECT-TYPE
    SYNTAX                  INTEGER ( 1..1024 )
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        ”This defines the value of the call retransmit timer, in seconds.”
    ::= { x25TTEntry 3 }

x25TTT22Timer  OBJECT-TYPE
```

```
    SYNTAX              INTEGER ( 1..1024 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This defines the value of the reset retransmit timer, in seconds.”
    ::= { x25TTEntry 4 }

x25TTT23Timer  OBJECT-TYPE
    SYNTAX              INTEGER ( 1..1024 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This defines the value of the clear retransmit timer, in seconds.”
    ::= { x25TTEntry 5 }

x25TTR20Limit  OBJECT-TYPE
    SYNTAX              INTEGER ( 1..255 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This defines the maximum value of the restart retransmit count, when operating as a
        DTE.”
    ::= { x25TTEntry 6 }

x25TTR22Limit  OBJECT-TYPE
    SYNTAX              INTEGER ( 1..255 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This defines the maximum value of the reset retransmit count, when operating as a
        DTE.”
    ::= { x25TTEntry 7 }

x25TTR23Limit  OBJECT-TYPE
    SYNTAX              INTEGER ( 1..255 )
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This defines the maximum value of the clear retransmit count, when operating as a
        DTE.”
    ::= { x25TTEntry 8 }
```

x25 Status Table tc "x25 Status Table "§

```
x25StatusTable  OBJECT-TYPE
    SYNTAX              SEQUENCE OF  X25StatusTableEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”This table holds status information for each X25 SVC  and for each interface.”
    ::= { x25 6 }

x25StatusTableEntry OBJECT-TYPE
    SYNTAX              X25StatusTableEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”Each entry keeps status information on each Interface/SVC.”
    INDEX                   { x25StatusIndex }
    ::= { x25StatusTable 1 }

X25StatusTableEntry ::= SEQUENCE {
    x25StatusIndex
        INTEGER,
    x25StatusIfStatus
        INTEGER,
    x25StatusSVCStatus
        INTEGER,
    x25StatusWinSize
        INTEGER,
    x25StatusPktSize
        INTEGER,
    x25StatusCauseLastInClear
        INTEGER,
    x25StatusDiagLastInClear
        INTEGER
    }
```

```
x25StatusIndex  OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This index pertains to the interface/SVC for which this entry is valid.”
    ::= { x25StatusTableEntry 1 }

x25StatusIfStatus OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This object defines the current status of the interface. More specifically, this is  the state
        of the logical channel number 0, which governs the state of the entire              SVC.”
    ::= { x25StatusTableEntry 2 }

x25StatusSVCStatus OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This object defines the current status of the SVC, more specifically, the state of the
        active logical channel. This object has no meaning if the state of the Interface is not
        r1PktLevelReady.”
    ::= { x25StatusTableEntry 3 }

x25StatusWinSize  OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the window size in operation for the current SVC. This may be less than  the
        Configured Window sizes depending on whether Flow Control is enabled,  and the result
        of the negotiation.”
    ::= { x25StatusTableEntry 4 }
```

```
x25StatusPktSize  OBJECT-TYPE
    SYNTAX                    INTEGER
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This is the Packet size in use for the current SVC. This may be less than the  Configured
        values, depending on whether Flow Control is enabled and if  negotiation has taken
        place.”
    ::= { x25StatusTableEntry 5 }

x25StatusCauseLastInClear  OBJECT-TYPE
    SYNTAX                    INTEGER( 0..255 )
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This object defines the Cause of the Last Incoming Clear Received on this  interface. It
        is in the form of a bit-map, the values and combinations of which are  too numerous to
        list  here. The values may be found by referring to the  appropriate section in the X25
        PLP standard from CCITT.”
    ::= { x25StatusTableEntry 6 }

x25StatusDiagLastInClear  OBJECT-TYPE
    SYNTAX                    INTEGER( 0..255 )
    ACCESS                    read-only
    STATUS                    mandatory

    DESCRIPTION
        ”This object defines the Diagnostic of the Last Incoming Clear Received on this
        interface. It is in the form of a bit-map, the values and combinations of which are  too
        numerous to list  here. The values may be found by referring to the  appropriate section
        in the X25 PLP standard from CCITT.”
    ::= { x25StatusTableEntry 7 }
```

x25 Statistics Tabletc "x25 Statistics Table"§

```
x25StatsTable  OBJECT-TYPE
    SYNTAX                    SEQUENCE OF  X25StatsTableEntry
    ACCESS                    not-accessible
    STATUS                    mandatory
    DESCRIPTION
        ”This table holds all statistical data for each x25 SVC . There are two entries in  this
        table, for a 4942.”
    ::= { x25 7 }

x25StatsTableEntry OBJECT-TYPE
    SYNTAX                  X25StatsTableEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        ”Each entry in the Table includes all the statistical objects for each interface.”
    INDEX                       { x25STSVCIndex }
    ::= { x25StatsTable 1 }

X25StatsTableEntry ::= SEQUENCE {
    x25STSVCIndex
        Counter,
    x25STDataPktsTx
        Counter,
    x25STDataPktsRx
        Counter,
    x25STConnectReqPktsTx
        Counter,
    x25STIncomingCallPktsRx
        Counter,
    x25STClearReqPktsTx
        Counter,
    x25STClearIndPktsRx
        Counter,
    x25STResetReqPktsTx
        Counter,
    x25STResetIndPktsRx
        Counter,
    x25STRestartReqPktsTx
        Counter,
    x25STRestartIndPktsRx
        Counter
    }

x25STSVCIndex  OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS              mandatory

    DESCRIPTION
        ”This index pertains to the interface for which this entry is valid.”
    ::= { x25StatsTableEntry 1 }

x25STDataPktsTx  OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the no. of Data Pkts transmitted on this interface.”
    ::= { x25StatsTableEntry 2 }
```

```
x25STDataPktsRx  OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the number of Data Pkts received on this interface.”
    ::= { x25StatsTableEntry 3 }

x25STConnectReqPktsTx  OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the number of Connect Request Pkts transmitted on this interface.”
    ::= { x25StatsTableEntry 4 }

x25STIncomingCallPktsRx  OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the number of incoming Call request Pkts received  on this interface.”
    ::= { x25StatsTableEntry 5 }

x25STClearReqPktsTx  OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the number of Clear Request Pkts transmitted on this interface.”
    ::= { x25StatsTableEntry 6 }

x25STClearIndPktsRx  OBJECT-TYPE
    SYNTAX                  Counter
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        ”This is the number of Clear Indication Pkts received on this interface.”
    ::= { x25StatsTableEntry 7 }
```

```
x25STResetReqPktsTx  OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the number of Reset Request Pkts transmitted on this interface.”
    ::= { x25StatsTableEntry 8 }

x25STResetIndPktsRx  OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the number of Reset Indication Pkts received on this interface.”
    ::= { x25StatsTableEntry 9 }

x25STRestartReqPktsTx  OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the number of Restart Request Pkts transmitted on this interface.”
    ::= { x25StatsTableEntry 10 }

x25STRestartIndPktsRx  OBJECT-TYPE
    SYNTAX              Counter
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        ”This is the number of Restart Indication Pkts received on this interface.”
    ::= { x25StatsTableEntry 11 }
```

Brdg common grouptc "Brdg common group"§

ACCESSPolicy mib objectstc "ACCESSPolicy mib objects"§

```
bcomBpduTimer OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”The time in seconds between the generation of  bridge BPDU’s.  This is the current
        working value.”
    ::=  { brdgcommon 1 }

bcomDefaultBpduTimer OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object is non-functioning.”
    ::=  { brdgcommon 2 }

snmpAccessPolicyObject    OBJECT IDENTIFIER ::= {brdgcommon 3 }
```

```
snmpAccessPolicyTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF SnmpAccessPolicyEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”This table is a list of entries which are used to authenticate PDUs received by the
        station. This table contains a maximum of 8 entries.”
    ::=  { snmpAccessPolicyObject 1 }

snmpAccessPolicyEntry OBJECT-TYPE
    SYNTAX              SnmpAccessPolicyEntry
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        ”This is an entry in the snmpAccessPolicyTable.”
    INDEX                   { accessPolicyIndex }
    ::=  { snmpAccessPolicyTable 1 }

SnmpAccessPolicyEntry ::= SEQUENCE {
    ACCESSPolicyIndex
        INTEGER,
    communityName
        OCTET STRING (SIZE(0..22)),
    ACCESSMode
        INTEGER,
    snmpAccessPolicyType
        INTEGER
    }

ACCESSPolicyIndex OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This index is used as the instance for a Get or Set operation on the table.”
    ::=  { snmpAccessPolicyEntry 1 }

communityName OBJECT-TYPE
    SYNTAX              OCTET STRING ( SIZE (0..22))
```

```
ACCESS          read-write
STATUS          mandatory

DESCRIPTION
    ”This object is a string, against which the community field in a received PDU is
    compared. If there exists no community name in the table, with which the received
    community field forms a match, authentication of the PDU fails.”
::=  { snmpAccessPolicyEntry 2 }
```

```
accessMode OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This is a bit significant field, as follows:

        bit 0           readAccess
        bit 1           writeAccess

    Setting the individual bit to 1 enables the type of access, setting it to 0 disables it.”
    ::=  { snmpAccessPolicyEntry 3 }

snmpAccessPolicyType OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        ”This object determines what action is to be performed on a particular entry. A value of
        valid causes the entry to be added to the table, while a value of invalid causes the entry to
        be deleted.”
    ::=  { snmpAccessPolicyEntry 4 }

snmpAccessPolicyAction OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              write-only
    STATUS              mandatory

    DESCRIPTION
        ”This object is used to manipulate the table as a whole. It may take the following values:

        clearTable      1"

    ::=  { snmpAccessPolicyObject 2 }
```

S3snmpAgent grouptc "S3snmpAgent group"§

```
s3AgentType OBJECT-TYPE
    SYNTAX              INTEGER {
        other (1),             — none of the following
        m3313 (2),             — AUI ethernet NMM w/ RS232 port
        m3313M (3),            — AUI enet NMM w/ RS232 port + modem
        m3314-ST (4),          — FOIRL ethernet NMM w/ RS232
        m3314M-ST (5),         — FOIRL ethernet NMM w/ RS232 + modem
        m2310 (6),             — Model 2310
        m3512 (7),             — T.R. NMM w/ STP ring in/ring out
        m3514 (8),             — T.R. NMM w/ FOIRL ring in/ring out
        m332X (9),             — Low Speed Local Bridge
        m3356 (10),            — Ethernet Remote Bridge
        m332XS (11),           — High Speed Local Bridge
        m3910S (12)            — FDDI Super NMM
    }
    ACCESS                     read-only
    STATUS                     mandatory

    DESCRIPTION
                "The 3000 series agent's Module type."
    ::= {s3AgentSw  1 }

s3AgentFwVer OBJECT-TYPE
    SYNTAX                     INTEGER
    ACCESS                     read-only
    STATUS                     mandatory

    DESCRIPTION
        "The agent's module firmware version 0=Rev A,1=Rev B,etc."
    ::= { s3AgentSw 2 }

s3AgentSwMajorVer OBJECT-TYPE
    SYNTAX                     INTEGER
    ACCESS                     read-only
    STATUS                     mandatory

    DESCRIPTION
        "The bridges major software version number (ie.represents major.minor or 3.0).  Object
        is in decimal."
    ::= { s3AgentSw 3 }

s3AgentSwMinorVer OBJECT-TYPE
    SYNTAX                     INTEGER
    ACCESS                     read-only
    STATUS                     mandatory
    DESCRIPTION
        "The bridges minor software version number (ie. represents major.minor or 3.0).  Object
        is in decimal."
    ::= { s3AgentSw 4 }

s3AgentBootProtocol OBJECT-TYPE
    SYNTAX                      INTEGER {
        other(1),               — boot protocol not listed here
        bootp-tftp(2),
        tftp-only(3),
        ieee802-1e(4),
        proprietary(5),         — supported proprietary boot protocol
        dll-download(6)         — direct async binary download (Retix)
    }
    ACCESS                      read-only
    STATUS                      mandatory

    DESCRIPTION
        "Boot protocol used to load the module with its software."
    ::= {s3AgentSw  5 }

s3AgentBootFile OBJECT-TYPE
    SYNTAX                      OCTET STRING (SIZE(64))
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        "The name of the boot file that is sent to the file server during boot time.  The actual boot
        protocol used to retreive the file is determined by s3AgentBootProtocol. This object
        shares its data/functionality with 'downloadFilename {boot 3}."
    ::= { s3AgentSw 6 }

s3AgentAuthTrap OBJECT-TYPE
    SYNTAX                      INTEGER {
        disable(1),             — disable authentication traps
        enable(2)               — enable authentication traps
    }
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        "Enable or disable the use of authentication error trap generation. This object shares it
        data/functionality with :

        snmpEnableAuthenTraps {snmp 30}and
        authenticationtrapStatus {station 14} "

    ::= { s3AgentSw 7 }

s3AgentLocation OBJECT-TYPE
    SYNTAX                  OCTET STRING (SIZE(64))
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        "This is a informational string that could be used to show the physical location (i.e. area)
        of the bridge or concentrator. This object shares its functionality/data with 'sysLocation'
        {system 6}"
    ::= { s3AgentSw  8 }

s3AgentMibLevel OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory
    DESCRIPTION
        “Used to determine current MIB release supported by agent.  The encoding of this object
        is of the following form: xyy, where x is the major version number and yy is the minor
        version number.  Thus, version 3.0 would be represented as 300,  version 3.01 would be
        301 and 3.1 would be 310.  Object is in decimal.”
    ::= { s3AgentSw  9 }

s3AgentFeatureLevel OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “Used to determine feature level of the agent.  The encoding of this object is of the
        following form: xyy, where x is the major version number and yy is the minor version
        number.  Thus, feature level version 1.0 would be represented as 100, version 1.01 would
        be 101 and 1.1 would be 110. Object is in decimal.”
    ::= { s3AgentSw  10 }

s3AgentMySlotID OBJECT-TYPE
    SYNTAX              INTEGER
    ACCESS              read-only
    STATUS              mandatory

    DESCRIPTION
        “Slot where this agent currently resides.  The Remote bridge hardware is unable to
        determine this and will always return zero.”
    ::= { s3AgentSw 11 }

s3AgentUnAuthIP OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              not-accessible
    STATUS              mandatory

    DESCRIPTION
        “This object contains the IP address of the last station that tried to access this agent with
        an invalid community string.  This object is only used as a VarBind in an Authentication
        Trap PDU.”
    ::= { s3AgentSw  12 }

s3AgentUnAuthComm OBJECT-TYPE
```

```
SYNTAX          OCTET STRING (SIZE(20))
ACCESS          not-accessible
STATUS          mandatory

DESCRIPTION
    “This object contains the community string of the last unauthenticated attempt to access
    this agent.  This object is only used as a VarBind in an Authentication Trap PDU.”
::= { s3AgentSw  13 }
```

```
s3AgentIpAddr OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The bridge's administrative IP address.  The current operational IP address can be
        obtained by the ipAdEntAddr entry in the ipAddrTable."
    ::= { s3AgentIpProtocol 1 }

s3AgentNetMask OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The bridge's administrative subnet mask."
    ::= { s3AgentIpProtocol 2 }

s3AgentDefaultGateway OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The bridge's administrative default gateway IP address."
    ::= { s3AgentIpProtocol 3 }

s3AgentBootServerAddr OBJECT-TYPE
    SYNTAX              IpAddress
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        "The IP address to which the requests for boot files are sent. The protocol used to
        retrieve the boot ffiles is determined by objects3AgentBootProtocol (section 2).This
        object shares its functionality and data with 'bootserverIpAddress' {boot 4}"
    ::= { s3AgentIpProtocol 4 }

s3AgentTrapReceiverTable OBJECT-TYPE
    SYNTAX              SEQUENCE OF S3AgentTrapReceiverEntry
    ACCESS              not-accessible
    STATUS                  mandatory

    DESCRIPTION
        “Table containing list of NMS’s that are to receive traps generated by this bridge.”
    ::= { s3SnmpAgent 3 }

s3AgentTrapReceiverEntry OBJECT-TYPE
    SYNTAX                  S3AgentTrapReceiverEntry
    ACCESS                  not-accessible
    STATUS                  mandatory

    DESCRIPTION
        “A destination address and community string to a particular trap server, along with a trap
        filter and entry aging capability.”
    INDEX                       { s3AgentTrapRcvrNetAddress }
    ::= { s3AgentTrapReceiverTable 1 }

S3AgentTrapReceiverEntry ::= SEQUENCE {
    s3AgentTrapRcvrStatus
        INTEGER,
    s3AgentTrapRcvrNetAddress
        IpAddress,
    s3AgentTrapRcvrComm
        OCTET STRING (SIZE(20)),
    s3AgentTrapRcvrAgeTime
        TimeTicks}

s3AgentTrapRcvrStatus OBJECT-TYPE
    SYNTAX                  INTEGER {
        other (1),          — none of the following
        valid (2),          — a valid address
        invalid (3)         — an invalid address
    }
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        “Setting this object to the value invalid(3) has the effect of invalidating the
        corresponding entry in the s3AgentTrapReceiverTable.  That is, it effectively
        disassociates the address identified with the entry by removing the entry from the table.
        On a read the value of valid(2) should be returned if the entry contains a valid record, or
        invalid(3) should be returned if the entry contains an invalid record.”
    ::= { s3AgentTrapReceiverEntry 1 }
```

```
s3AgentTrapRcvrNetAddress OBJECT-TYPE
    SYNTAX                      OCTET STRING (SIZE(4))
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        “IP address for trap server that is to receive the trap.”
    ::= { s3AgentTrapReceiverEntry  2 }

s3AgentTrapRcvrComm OBJECT-TYPE
    SYNTAX                      OCTET STRING (SIZE(20))
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        “Community string used for traps sent to this trap receiver.”
    ::= { s3AgentTrapReceiverEntry  3 }

s3AgentTrapRcvrAgeTime OBJECT-TYPE
    SYNTAX                      TimeTicks
    ACCESS                      read-write
    STATUS                      mandatory

    DESCRIPTION
        “This is the time interval used to age entries out of the trap receiver table.  The default
        value if not specified will be 0, or infinite, never to be aged out.”
    ::= { s3AgentTrapReceiverEntry  4 }

s3AgentStatus OBJECT-TYPE
    SYNTAX                      INTEGER {
        offline(1),             — the agent is offline
        online(2)               — the agent is online
    }
    ACCESS                      read-only
    STATUS                      mandatory

    DESCRIPTION
        “This indicates whether the bridge is online(2) and operating, or offline(1) (ie: not
        connected to the concentrator backplane). The Remote Bridge hardware is unable to
        determine this and will always return ‘online’”
    ::= { s3AgentHw 1 }
```

```
s3AgentMdaHwVer OBJECT-TYPE
    SYNTAX                  INTEGER
    ACCESS                  read-only
    STATUS                  mandatory

    DESCRIPTION
        "The network management module's MDA hardware version number (0=Rev. A, 1=Rev.
        B, etc.). The remote bridge hardware is unable to determine this and will always return
        zero"
    ::= { s3AgentHw 2 }

s3AgentMode OBJECT-TYPE
    SYNTAX                  INTEGER {
        primary(1),         — in primary mode
        secondary(2),       — in secondary mode
        other(3)            — the other values do not apply
    }
    ACCESS             read-write
    STATUS                  mandatory

    DESCRIPTION
        "This indicates whether the module is operating in primary(1) or secondary(2) mode.
        This is not relevant to the remote bridge which will always return 'other'"
    ::= { s3AgentHw 3 }
```

```
s3AgentReset OBJECT-TYPE
    SYNTAX                  INTEGER {
        noReset(1),         — not resetting
        reset(2)            — reset
    }
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        " Writing a reset(2) to this object will reset the bridge (stop refreshinng the watch dog)."
    ::= { s3AgentHw 4 }

s3AgentRestart OBJECT-TYPE
    SYNTAX                  INTEGER {
        noRestart(1),       — not restarting
        restart(2)          — restart
    }
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        "Writing a reset(2) to this object will reset the bridge (stop refreshing the watch dog)."
    ::= { s3AgentHw 5 }

s3AgentBootMode OBJECT-TYPE
    SYNTAX                  INTEGER {
        eeprom(1),          — get boot info from eeprom
        other(2)            — or from a boot protocol
    }
    ACCESS                  read-write
    STATUS                  mandatory

    DESCRIPTION
        "This parameter will be non-functioning, and will always return 'eeprom'.  Attempting to
        write any other value will return a 'bad-value error."
    ::= { s3AgentHw 6 }

s3AgentWriteEeprom OBJECT-TYPE
    SYNTAX                  INTEGER {
        noWriteEeprom(1),   — not writing EEPROM
        writeEeprom(2)      — write EEPROM
    }
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “This object is non-functioning, any attempt to set the value ‘writeEeprom’ will return a
        ‘bad-value’ error, and it will always be read as ‘noWrtieEeprom’.”
    ::= { s3AgentHw 7 }

s3AgentBaudRate OBJECT-TYPE
    SYNTAX              Gauge
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “Specifies the baud rate in bits per second of the console port of the bridge.”
    ::= { s3AgentHw 8 }

s3AgentInitString OBJECT-TYPE
    SYNTAX              OCTET STRING (SIZE(64))
    ACCESS              read-write
    STATUS              mandatory

    DESCRIPTION
        “This parameter will be non-functioning, when read will always return a null string, and
        any attempt to write a non-null string to it will return a ‘gen-err’.”
    ::= { s3AgentHw 9 }

END
```

Trapstc "Traps"§

generic traps:tc "generic traps\:"§

```
coldStart TRAP-TYPE
    ENTERPRISE  snmp

    DESCRIPTION
        “A coldStart trap signifies that the sending protocol entity is reinitializing itself such that
        the agent’s configuration or the protocol entity implementation may be altered.”
    ::= 0
```

```
warmStart TRAP-TYPE
    ENTERPRISE  snmp

    DESCRIPTION
        “A warmStart trap signifies that the sending protocol entity is reinitializing itself such
        that neither the agent configuration nor the protocol entity implementation is altered.”
    ::= 1

linkDown TRAP-TYPE
    ENTERPRISE  snmp
    VARIABLES  { ifIndex }

    DESCRIPTION
        “A linkDown trap signifies that the sending protocol entity recognizes a failure in one of
        the communication links represented in the agent’s configuration.”
    ::= 2

linkUp TRAP-TYPE
    ENTERPRISE  snmp
    VARIABLES  { ifIndex }

    DESCRIPTION
        “A linkUp trap signifies that the sending protocol entity recognizes that one of the
        communication links represented in the agent’s configuration has come up.”
    ::= 3

authenticationFailure TRAP-TYPE
    ENTERPRISE  snmp

    DESCRIPTION
        “An authenticationFailure trap signifies that the sending protocol entity is the addressee
        of a protocol message that is not properly authenticated.  While implementations of the
        SNMP must be capable of generating this trap, they must also be capable of suppressing
        the emission of such traps via an implementation- specific mechanism.”
    ::= 4

synoptics  OBJECT IDENTIFIER ::= { enterprises 45 }

topologyChangeTrap TRAP-TYPE
    ENTERPRISE  synoptics

    DESCRIPTION
        “A topologyChangeTrap signifies a topology change has been detected by the bridge.
        This is generated by the root bridge of the network following a topology change when
        the network has stabilised,  thus preventing a flood of traps.”
    ::= 2
```