Definitions of Managed Objects
for Parallel-printer-like Hardware Devices

Status of this Memo

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

1. Abstract

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

2. The Network Management Framework

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

- **RFC 1155** which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.
- **RFC 1212** defines a more concise description mechanism, which is wholly consistent with the SMI.
- **RFC 1156** which defines MIB-I, the core set of managed objects for the Internet suite of protocols. **RFC 1213**, defines MIB-II, an evolution of MIB-I based on implementation experience and new operational requirements.
- **RFC 1157** which defines the SNMP, the protocol used for network access to managed objects.

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

3. Objects

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB.
Objects in the MIB are defined using the subset of Abstract Syntax Notation One (ASN.1) [7] defined in the SMI. In particular, each object has a name, a syntax, and an encoding. The name is an object identifier, an administratively assigned name, which specifies an object type.

The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the OBJECT DESCRIPTOR, to also refer to the object type.

The syntax of an object type defines the abstract data structure corresponding to that object type. The ASN.1 language is used for this purpose. However, the SMI [3] purposely restricts the ASN.1 constructs which may be used. These restrictions are explicitly made for simplicity.

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

The SMI specifies the use of the basic encoding rules of ASN.1 [8], subject to the additional requirements imposed by the SNMP.

3.1. Format of Definitions

Section 5 contains the specification of all object types contained in this MIB module. The object types are defined using the conventions defined in the SMI, as amended by the extensions specified in [9,10].

4. Overview

The Parallel-printer-like Hardware Device MIB applies to interface ports that might logically support the Interface MIB, a Transmission MIB, or the Character MIB (most likely the latter). The most common example is a Centronics or Data Products type parallel printer port.

The Parallel-printer-like MIB is one of a set of MIBs designed for complementary use. At this writing, the set comprises:
The RS-232-like MIB and the Parallel-printer-like MIB represent the physical layer, providing service to higher layers such as the Character MIB or PPP MIB. Further MIBs may appear above these.

The following diagram shows two possible "MIB stacks", each using the RS-232-like MIB.

```
    -----------------        |  Standard MIB   |
    |   Telnet MIB    |        | Interface Group |
    |-----------------|        |-----------------|
    |  Character MIB  |        |     PPP MIB     |
    |-----------------|        |-----------------|
    | RS-232-like MIB |        | RS-232-like MIB |
    '-----------------'        '-----------------'
```

The intent of the model is for the physical-level MIBs to represent the lowest level, regardless of the higher level that may be using it. In turn, separate higher level MIBs represent specific applications, such as a terminal (the Character MIB) or a network connection (the PPP MIB).

The Parallel-printer-like MIB is mandatory for all systems that have such a hardware port supporting services managed through some other MIB, for example, the Character MIB.

The Parallel-printer-like MIB includes multiple similar types of hardware, and as a result contains objects not applicable to all of those types. Such objects are in a separate branch of the MIB, which is required when applicable and otherwise absent.

The Parallel-printer-like MIB includes Centronics, Data Products, and other parallel physical links with a similar set of control signals.

The MIB contains objects that relate to physical layer connections. Such connections may provide interesting hardware signals (other than for basic data transfer), such as Power and PaperOut.

The MIB comprises one base object and three tables, detailed
in the following sections. The tables contain objects for
ports and input and output control signals.

5. Definitions

```
RFC1318-MIB DEFINITIONS ::= BEGIN

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

-- this is the MIB module for Parallel-printer-like
-- hardware devices
para OBJECT IDENTIFIER ::= { transmission 34 }

-- the generic Parallel-printer-like group

-- Implementation of this group is mandatory for all
-- systems that have Parallel-printer-like hardware
-- ports supporting higher level services such as
-- character streams

paraNumber OBJECT-TYPE
  SYNTAX INTEGER
  ACCESS read-only
  STATUS mandatory
  DESCRIPTION
    "The number of ports (regardless of their current
    state) in the Parallel-printer-like port table."
  ::= { para 1 }

-- the Parallel-printer-like Port table

paraPortTable OBJECT-TYPE
  SYNTAX SEQUENCE OF ParaPortEntry
  ACCESS not-accessible
  STATUS mandatory
  DESCRIPTION
    "A list of port entries. The number of entries is
given by the value of paraNumber."
  ::= { para 2 }
```
paraPortEntry OBJECT-TYPE
SYNTAX ParaPortEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Status and parameter values for a port."
INDEX { paraPortIndex }
::= { paraPortTable 1 }

ParaPortEntry ::= 
  SEQUENCE {
    paraPortIndex
      INTEGER,
    paraPortType
      INTEGER,
    paraPortInSigNumber
      INTEGER,
    paraPortOutSigNumber
      INTEGER
  }

paraPortIndex OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"A unique value for each port. Its value ranges between 1 and the value of paraNumber. By convention and if possible, hardware port numbers map directly to external connectors. The value for each port must remain constant at least from one re-initialization of the network management agent to the next."
::= { paraPortEntry 1 }

paraPortType OBJECT-TYPE
SYNTAX INTEGER {
  other(1),
  centronics(2),
  dataproducts(3)
}
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The port’s hardware type."
::= { paraPortEntry 2 }

paraPortInSigNumber OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of input signals for the port in the input signal table (paraPortInSigTable). The table contains entries only for those signals the software can detect."
::= { paraPortEntry 3 }

paraPortOutSigNumber OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of output signals for the port in the output signal table (paraPortOutSigTable). The table contains entries only for those signals the software can assert."
::= { paraPortEntry 4 }

-- the Input Signal table

paraInSigTable OBJECT-TYPE
SYNTAX SEQUENCE OF ParaInSigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"A list of port input control signal entries."
::= { paraInSigTable 1 }

ParaInSigEntry OBJECT-TYPE
SYNTAX ParaInSigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Input control signal status for a hardware port."
INDEX { paraInSigPortIndex, paraInSigName }
::= { paraInSigTable 1 }

ParaInSigEntry ::= SEQUENCE {
    paraInSigPortIndex
        INTEGER,
    paraInSigName
        INTEGER,
    paraInSigState
        INTEGER}
INTEGER,
    paraInSigChanges
Counter

paraInSigPortIndex OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
   "The value of paraPortIndex for the port to which
    this entry belongs."
 ::= { paraInSigEntry 1 }

paraInSigName OBJECT-TYPE
SYNTAX INTEGER { power(1), online(2), busy(3),
               paperout(4), fault(5) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
   "Identification of a hardware signal."
 ::= { paraInSigEntry 2 }

paraInSigState OBJECT-TYPE
SYNTAX INTEGER { none(1), on(2), off(3) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
   "The current signal state."
 ::= { paraInSigEntry 3 }

paraInSigChanges OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
   "The number of times the signal has changed from
    'on' to 'off' or from 'off' to 'on'.''"
 ::= { paraInSigEntry 4 }

-- the Output Signal table

paraOutSigTable OBJECT-TYPE
SYNTAX SEQUENCE OF ParaOutSigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"A list of port output control signal entries."
::= { para 4 }

paraOutSigEntry OBJECT-TYPE
SYNTAX ParaOutSigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION
"Output control signal status for a hardware port."
INDEX { paraOutSigPortIndex, paraOutSigName }
::= { paraOutSigTable 1 }

ParaOutSigEntry ::= SEQUENCE {
  paraOutSigPortIndex          INTEGER,
  paraOutSigName               INTEGER,
  paraOutSigState              INTEGER,
  paraOutSigChanges            Counter
}

paraOutSigPortIndex OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The value of paraPortIndex for the port to which this entry belongs."
::= { paraOutSigEntry 1 }

paraOutSigName OBJECT-TYPE
SYNTAX INTEGER { power(1), online(2), busy(3),
  paperout(4), fault(5) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
"Identification of a hardware signal."
::= { paraOutSigEntry 2 }

paraOutSigState OBJECT-TYPE
SYNTAX INTEGER { none(1), on(2), off(3) }
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The current signal state."
::= { paraOutSigEntry 3 }

paraOutSigChanges OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of times the signal has changed from 'on' to 'off' or from 'off' to 'on'."
::= { paraOutSigEntry 4 }

END

6. Acknowledgements

Based on several private MIBs, this document was produced by the Character MIB Working Group:

Anne Ambler, Spider
Charles Bazaar, Emulex
Christopher Bucci, Datability
Anthony Chung, Hughes LAN Systems
George Conant, Xyplex
John Cook, Chipcom
James Davin, MIT-LCS
Shawn Gallagher, DEC
Tom Grant, Xylogics
Frank Huang, Emulex
David Jordan, Emulex
Satish Joshi, SynOptics
Frank Kastenholz, Clearpoint
Ken Key, University of Tennessee
Jim Kinder, Fibercom
Rajeev Kochhar, 3Com
John LoVerso, Xylogics
Keith McCloghrie, Hughes LAN Systems
Donald Merritt, BRL
David Perkins, 3Com
Jim Reinstedler, Ungerman-Bass
Marshall Rose, PSI
Ron Strich, SSDS
Dean Throop, DG
Bill Townsend, Xylogics
Jesse Walker, DEC
David Waitzman, BBN
Bill Westfield, cisco
7. References


8. Security Considerations

Security issues are not discussed in this memo.
9. Author’s Address

Bob Stewart
Xyplex, Inc.
330 Codman Hill Road
Boxborough, MA 01719

Phone: (508) 264-9900
EMail: rlstewart@eng.xyplex.com