

Network Working Group
Request for Comments: 1659
Obsoletes: 1317
Category: Standards Track

B. Stewart
Xyplex, Inc.
July 1994

Definitions of Managed Objects for RS-232-like Hardware Devices using SMIV2

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Introduction	1
2. The SNMPv2 Network Management Framework	1
2.1 Object Definitions	2
3. Overview	2
3.1 Relationship to Interface MIB	3
4. Definitions	3
5. Acknowledgements	20
6. References	20
7. Security Considerations	21
8. Author's Address	21

1. Introduction

This memo defines an extension to the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it defines objects for the management of RS-232-like devices.

2. The SNMPv2 Network Management Framework

The SNMPv2 Network Management Framework consists of four major components. They are:

- o RFC 1442 [1] which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.
- o STD 17, RFC 1213 [2] defines MIB-II, the core set of managed objects for the Internet suite of protocols.

- o RFC 1445 [3] which defines the administrative and other architectural aspects of the framework.
- o RFC 1448 [4] which defines the protocol used for network access to managed objects.

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

2.1. Object Definitions

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) defined in the SMI. In particular, each object type is named by an OBJECT IDENTIFIER, an administratively assigned name. 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 descriptor, to refer to the object type.

3. Overview

The RS-232-like Hardware Device MIB applies to interface ports that might logically support the Interface MIB, a Transmission MIB, or the Character MIB. The most common example is an RS-232 port with modem signals.

The RS-232-like Hardware Device MIB is mandatory for all systems that have such a hardware port supporting services managed through some other MIB.

The MIB includes multiple similar types of hardware, and as a result contains objects not applicable to all of those types. The compliance definitions herein thus have a general group for all implementations, and separate groups for the different types of ports, such as asynchronous and synchronous.

The RS-232-like Hardware Port MIB includes RS-232, RS-422, RS-423, V.35, and other asynchronous or synchronous, serial 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 RNG and DCD. Hardware ports also have such attributes as speed and bits per character.

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

3.1. Relationship to Interface MIB

The RS-232-like MIB is one of many MIBs designed for layered use as described in the Interface MIB [5]. In most implementations where it is present, it will be in the lowest interface sublayer, that is, the RS-232-like MIB represents the physical layer, providing service to higher layers such as the Character MIB [6] or PPP MIB [7].

The Interface MIB's ifTestTable and ifRcvAddressTable are not relevant to the RS-232-like MIB.

The RS-232-like MIB is relevant for ifType values rs232(33), v35(45), and perhaps others.

The RS-232-like MIB requires the conformance groups ifGeneralGroup, and ifFixedLengthGroup.

The value of ifSpeed is the same as rs232PortOutSpeed.

Usefulness of error counters in this MIB depends on the octet counters in ifFixedLengthGroup.

4. Definitions

RS-232-MIB DEFINITIONS ::= BEGIN

IMPORTS

```
MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
Counter32, Integer32
    FROM SNMPv2-SMI
InterfaceIndex
    FROM IF-MIB
transmission
    FROM RFC1213-MIB
MODULE-COMPLIANCE, OBJECT-GROUP
    FROM SNMPv2-CONF;
```

rs232 MODULE-IDENTITY

```
LAST-UPDATED "9405261700Z"
ORGANIZATION "IETF Character MIB Working Group"
CONTACT-INFO
    "          Bob Stewart
    Postal: Xyplex, Inc.
```

295 Foster Street
Littleton, MA 01460

Tel: 508-952-4816

Fax: 508-952-4887

E-mail: rlstewart@eng.xyplex.com"

DESCRIPTION

"The MIB module for RS-232-like hardware devices."

::= { transmission 33 }

-- Generic RS-232-like information

rs232Number OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of ports (regardless of their current
state) in the RS-232-like general port table."

::= { rs232 1 }

-- RS-232-like General Port Table

rs232PortTable OBJECT-TYPE

SYNTAX SEQUENCE OF Rs232PortEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A list of port entries. The number of entries is
given by the value of rs232Number."

::= { rs232 2 }

rs232PortEntry OBJECT-TYPE

SYNTAX Rs232PortEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Status and parameter values for a port."

INDEX { rs232PortIndex }

::= { rs232PortTable 1 }

Rs232PortEntry ::=

SEQUENCE {

rs232PortIndex

InterfaceIndex,

rs232PortType

```

        INTEGER,
rs232PortInSigNumber
        Integer32,
rs232PortOutSigNumber
        Integer32,
rs232PortInSpeed
        Integer32,
rs232PortOutSpeed
        Integer32,
rs232PortInFlowType
        INTEGER,
rs232PortOutFlowType
        INTEGER
    }

```

rs232PortIndex OBJECT-TYPE

SYNTAX InterfaceIndex

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of ifIndex for the port. 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."

::= { rs232PortEntry 1 }

rs232PortType OBJECT-TYPE

SYNTAX INTEGER { other(1), rs232(2), rs422(3),
rs423(4), v35(5), x21(6) }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The port's hardware type."

::= { rs232PortEntry 2 }

rs232PortInSigNumber OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of input signals for the port in the input signal table (rs232PortInSigTable). The table contains entries only for those signals the software can detect and that are useful to observe."

::= { rs232PortEntry 3 }

rs232PortOutSigNumber OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of output signals for the port in the output signal table (rs232PortOutSigTable). The table contains entries only for those signals the software can assert and that are useful to observe."

::= { rs232PortEntry 4 }

rs232PortInSpeed OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The port's input speed in bits per second. Note that non-standard values, such as 9612, are probably not allowed on most implementations."

::= { rs232PortEntry 5 }

rs232PortOutSpeed OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The port's output speed in bits per second. Note that non-standard values, such as 9612, are probably not allowed on most implementations."

::= { rs232PortEntry 6 }

rs232PortInFlowType OBJECT-TYPE

SYNTAX INTEGER { none(1), ctsRts(2), dsrDtr(3) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The port's type of input flow control. 'none' indicates no flow control at this level. 'ctsRts' and 'dsrDtr' indicate use of the indicated hardware signals."

::= { rs232PortEntry 7 }

rs232PortOutFlowType OBJECT-TYPE

SYNTAX INTEGER { none(1), ctsRts(2), dsrDtr(3) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The port's type of output flow control. 'none'

```

        indicates no flow control at this level.
        'ctsRts' and 'dsrDtr' indicate use of the indicated
        hardware signals."
 ::= { rs232PortEntry 8 }

```

-- RS-232-like Asynchronous Port Table

```

rs232AsyncPortTable OBJECT-TYPE
    SYNTAX SEQUENCE OF Rs232AsyncPortEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of asynchronous port entries. Entries need
        not exist for synchronous ports."
    ::= { rs232 3 }

```

```

rs232AsyncPortEntry OBJECT-TYPE
    SYNTAX Rs232AsyncPortEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Status and parameter values for an asynchronous
        port."
    INDEX { rs232AsyncPortIndex }
    ::= { rs232AsyncPortTable 1 }

```

```

Rs232AsyncPortEntry ::=
    SEQUENCE {
        rs232AsyncPortIndex
            InterfaceIndex,
        rs232AsyncPortBits
            INTEGER,
        rs232AsyncPortStopBits
            INTEGER,
        rs232AsyncPortParity
            INTEGER,
        rs232AsyncPortAutobaud
            INTEGER,
        rs232AsyncPortParityErrs
            Counter32,
        rs232AsyncPortFramingErrs
            Counter32,
        rs232AsyncPortOverrunErrs
            Counter32
    }

```

rs232AsyncPortIndex OBJECT-TYPE

SYNTAX InterfaceIndex

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A unique value for each port. Its value is the same as rs232PortIndex for the port."

::= { rs232AsyncPortEntry 1 }

rs232AsyncPortBits OBJECT-TYPE

SYNTAX INTEGER (5..8)

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The port's number of bits in a character."

::= { rs232AsyncPortEntry 2 }

rs232AsyncPortStopBits OBJECT-TYPESYNTAX INTEGER { one(1), two(2),
oneAndHalf(3), dynamic(4) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The port's number of stop bits."

::= { rs232AsyncPortEntry 3 }

rs232AsyncPortParity OBJECT-TYPESYNTAX INTEGER { none(1), odd(2), even(3),
mark(4), space(5) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The port's sense of a character parity bit."

::= { rs232AsyncPortEntry 4 }

rs232AsyncPortAutobaud OBJECT-TYPE

SYNTAX INTEGER { enabled(1), disabled(2) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"A control for the port's ability to automatically sense input speed."

When rs232PortAutoBaud is 'enabled', a port may autobaud to values different from the set values for speed, parity, and character size. As a result a network management system may temporarily observe values different from what was previously set."

```
 ::= { rs232AsyncPortEntry 5 }

rs232AsyncPortParityErrs OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Total number of characters with a parity error,
        input from the port since system re-initialization
        and while the port state was 'up' or 'test'."
    ::= { rs232AsyncPortEntry 6 }

rs232AsyncPortFramingErrs OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Total number of characters with a framing error,
        input from the port since system re-initialization
        and while the port state was 'up' or 'test'."
    ::= { rs232AsyncPortEntry 7 }

rs232AsyncPortOverrunErrs OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Total number of characters with an overrun error,
        input from the port since system re-initialization
        and while the port state was 'up' or 'test'."
    ::= { rs232AsyncPortEntry 8 }

-- RS-232-like Synchronous Port Table

rs232SyncPortTable OBJECT-TYPE
    SYNTAX SEQUENCE OF Rs232SyncPortEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "A list of asynchronous port entries.  Entries need
        not exist for synchronous ports."
    ::= { rs232 4 }

rs232SyncPortEntry OBJECT-TYPE
    SYNTAX Rs232SyncPortEntry
    MAX-ACCESS not-accessible
    STATUS current
```

DESCRIPTION

"Status and parameter values for a synchronous port."

INDEX { rs232SyncPortIndex }
::= { rs232SyncPortTable 1 }

Rs232SyncPortEntry ::=

SEQUENCE {
 rs232SyncPortIndex
 InterfaceIndex,
 rs232SyncPortClockSource
 INTEGER,
 rs232SyncPortFrameCheckErrs
 Counter32,
 rs232SyncPortTransmitUnderrunErrs
 Counter32,
 rs232SyncPortReceiveOverrunErrs
 Counter32,
 rs232SyncPortInterruptedFrames
 Counter32,
 rs232SyncPortAbortedFrames
 Counter32,
 rs232SyncPortRole
 INTEGER,
 rs232SyncPortEncoding
 INTEGER,
 rs232SyncPortRTSControl
 INTEGER,
 rs232SyncPortRTSCTSDelay
 Integer32,
 rs232SyncPortMode
 INTEGER,
 rs232SyncPortIdlePattern
 INTEGER,
 rs232SyncPortMinFlags
 Integer32
}

rs232SyncPortIndex OBJECT-TYPE

SYNTAX InterfaceIndex

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A unique value for each port. Its value is the same as rs232PortIndex for the port."

::= { rs232SyncPortEntry 1 }

rs232SyncPortClockSource OBJECT-TYPE

SYNTAX INTEGER { internal(1), external(2), split(3) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"Source of the port's bit rate clock. 'split' means the transmit clock is internal and the receive clock is external."

::= { rs232SyncPortEntry 2 }

rs232SyncPortFrameCheckErrs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Total number of frames with an invalid frame check sequence, input from the port since system re-initialization and while the port state was 'up' or 'test'."

::= { rs232SyncPortEntry 3 }

rs232SyncPortTransmitUnderrunErrs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Total number of frames that failed to be transmitted on the port since system re-initialization and while the port state was 'up' or 'test' because data was not available to the transmitter in time."

::= { rs232SyncPortEntry 4 }

rs232SyncPortReceiveOverrunErrs OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Total number of frames that failed to be received on the port since system re-initialization and while the port state was 'up' or 'test' because the receiver did not accept the data in time."

::= { rs232SyncPortEntry 5 }

rs232SyncPortInterruptedFrames OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Total number of frames that failed to be received or transmitted on the port due to loss of modem signals since system re-initialization and while the port state was 'up' or 'test'."

::= { rs232SyncPortEntry 6 }

rs232SyncPortAbortedFrames OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Number of frames aborted on the port due to receiving an abort sequence since system re-initialization and while the port state was 'up' or 'test'."

::= { rs232SyncPortEntry 7 }

rs232SyncPortRole OBJECT-TYPE

SYNTAX INTEGER { dte(1), dce(2) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The role the device is playing that is using this port.

dte means the device is performing the role of data terminal equipment

dce means the device is performing the role of data circuit-terminating equipment."

DEFVAL { dce }

::= { rs232SyncPortEntry 8 }

rs232SyncPortEncoding OBJECT-TYPE

SYNTAX INTEGER { nrz(1), nrzi(2) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The bit stream encoding technique that is in effect for this port.

nrz for Non-Return to Zero encoding

nrzi for Non-Return to Zero Inverted encoding."

DEFVAL { nrz }

::= { rs232SyncPortEntry 9 }

rs232SyncPortRTSControl OBJECT-TYPE

SYNTAX INTEGER { controlled(1), constant(2) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The method used to control the Request To Send (RTS) signal.

controlled	when the DTE asserts RTS each time data needs to be transmitted and drops RTS at some point after data transmission begins.
------------	---

If rs232SyncPortRole is 'dte', the RTS is an output signal. The device will issue a RTS and wait for a CTS from the DCE before starting to transmit.

If `rs232SyncPortRole` is 'dce', the RTS is an input signal. The device will issue a CTS only after having received RTS and waiting the `rs232SyncPortRTSCTSDelay` interval.

```

        constant      when the DTE constantly asserts RTS."
DEFVAL { constant }
 ::= { rs232SyncPortEntry 10 }

```

```
DEFVAL { constant }
```

```
 ::= { rs232SyncPortEntry 10 }
```

rs232SyncPortRTSCTSDeLay OBJECT-TYPE

SYNTAX Integer32

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The interval (in milliseconds) that the DCE must wait after it sees RTS asserted before asserting CTS. This object exists in support of older synchronous devices that cannot recognize CTS within a certain interval after it asserts RTS."

```
DEFVAL { 0 }
```

```
 ::= { rs232SyncPortEntry 11 }
```

rs232SyncPortMode OBJECT-TYPE

```
SYNTAX INTEGER { fdx(1), hdx(2), simplex-receive(3),
                simplex-send(4) }
```

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The mode of operation of the port with respect to the direction and simultaneity of data transfer.

fdx when frames on the data link can be transmitted and received at the same time

hdx when frames can either be received from the data link or transmitted onto the data link but not at the same time.

simplex-receive when frames can only be received on this data link.

simplex-send when frames can only be sent on this data link."

DEFVAL { fdx }
::= { rs232SyncPortEntry 12 }

rs232SyncPortIdlePattern OBJECT-TYPE
SYNTAX INTEGER { mark(1), space(2) }
MAX-ACCESS read-write
STATUS current
DESCRIPTION
 "The bit pattern used to indicate an idle line."
DEFVAL { space }
::= { rs232SyncPortEntry 13 }

rs232SyncPortMinFlags OBJECT-TYPE
SYNTAX Integer32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
 "The minimum number of flag patterns this port needs in order to recognize the end of one frame and the start of the next. Plausible values are 1 and 2."
DEFVAL { 2 }
::= { rs232SyncPortEntry 14 }

-- Input Signal Table

rs232InSigTable OBJECT-TYPE
SYNTAX SEQUENCE OF Rs232InSigEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
 "A list of port input control signal entries implemented and visible to the software on the port, and useful to monitor."

```
::= { rs232 5 }
```

```
rs232InSigEntry OBJECT-TYPE
    SYNTAX Rs232InSigEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "Input control signal status for a hardware port."
    INDEX { rs232InSigPortIndex, rs232InSigName }
    ::= { rs232InSigTable 1 }
```

```
Rs232InSigEntry ::=
    SEQUENCE {
        rs232InSigPortIndex
            InterfaceIndex,
        rs232InSigName
            INTEGER,
        rs232InSigState
            INTEGER,
        rs232InSigChanges
            Counter32
    }
```

```
rs232InSigPortIndex OBJECT-TYPE
    SYNTAX InterfaceIndex
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of rs232PortIndex for the port to which
        this entry belongs."
    ::= { rs232InSigEntry 1 }
```

```
rs232InSigName OBJECT-TYPE
    SYNTAX INTEGER { rts(1), cts(2), dsr(3), dtr(4), ri(5),
                    dcd(6), sq(7), srs(8), srts(9),
                    scts(10), sdcd(11) }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Identification of a hardware signal, as follows:
```

rts	Request to Send
cts	Clear to Send
dsr	Data Set Ready
dtr	Data Terminal Ready
ri	Ring Indicator
dcd	Received Line Signal Detector
sq	Signal Quality Detector

```

        srs      Data Signaling Rate Selector
        srts     Secondary Request to Send
        scts     Secondary Clear to Send
        sdcld    Secondary Received Line Signal Detector
    "

```

REFERENCE

"EIA Standard RS-232-C, August 1969."

```
 ::= { rs232InSigEntry 2 }
```

rs232InSigState OBJECT-TYPE

```
 SYNTAX INTEGER { none(1), on(2), off(3) }
```

```
 MAX-ACCESS read-only
```

```
 STATUS current
```

```
 DESCRIPTION
```

"The current signal state."

```
 ::= { rs232InSigEntry 3 }
```

rs232InSigChanges OBJECT-TYPE

```
 SYNTAX Counter32
```

```
 MAX-ACCESS read-only
```

```
 STATUS current
```

```
 DESCRIPTION
```

"The number of times the signal has changed from
'on' to 'off' or from 'off' to 'on'."

```
 ::= { rs232InSigEntry 4 }
```

-- Output Signal Table

rs232OutSigTable OBJECT-TYPE

```
 SYNTAX SEQUENCE OF Rs232OutSigEntry
```

```
 MAX-ACCESS not-accessible
```

```
 STATUS current
```

```
 DESCRIPTION
```

"A list of port output control signal entries
implemented and visible to the software on the port,
and useful to monitor."

```
 ::= { rs232 6 }
```

rs232OutSigEntry OBJECT-TYPE

```
 SYNTAX Rs232OutSigEntry
```

```
 MAX-ACCESS not-accessible
```

```
 STATUS current
```

```
 DESCRIPTION
```

"Output control signal status for a hardware port."

```
 INDEX { rs232OutSigPortIndex, rs232OutSigName }
```

```
 ::= { rs232OutSigTable 1 }
```

```

Rs232OutSigEntry ::=
    SEQUENCE {
        rs232OutSigPortIndex
            InterfaceIndex,
        rs232OutSigName
            INTEGER,
        rs232OutSigState
            INTEGER,
        rs232OutSigChanges
            Counter32
    }

rs232OutSigPortIndex OBJECT-TYPE
    SYNTAX InterfaceIndex
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The value of rs232PortIndex for the port to which
        this entry belongs."
    ::= { rs232OutSigEntry 1 }

rs232OutSigName OBJECT-TYPE
    SYNTAX INTEGER { rts(1), cts(2), dsr(3), dtr(4), ri(5),
                    dcd(6), sq(7), srs(8), srts(9),
                    scts(10), sdcd(11) }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Identification of a hardware signal, as follows:

            rts    Request to Send
            cts    Clear to Send
            dsr    Data Set Ready
            dtr    Data Terminal Ready
            ri     Ring Indicator
            dcd    Received Line Signal Detector
            sq     Signal Quality Detector
            srs    Data Signaling Rate Selector
            srts   Secondary Request to Send
            scts   Secondary Clear to Send
            sdcd   Secondary Received Line Signal Detector
        "
    REFERENCE
        "EIA Standard RS-232-C, August 1969."
    ::= { rs232OutSigEntry 2 }

rs232OutSigState OBJECT-TYPE
    SYNTAX INTEGER { none(1), on(2), off(3) }

```

```
MAX-ACCESS read-only
STATUS current
DESCRIPTION
    "The current signal state."
::= { rs232OutSigEntry 3 }

rs232OutSigChanges OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of times the signal has changed from
        'on' to 'off' or from 'off' to 'on'."
    ::= { rs232OutSigEntry 4 }

-- conformance information

rs232Conformance OBJECT IDENTIFIER ::= { rs232 7 }

rs232Groups          OBJECT IDENTIFIER ::= { rs232Conformance 1 }
rs232Compliances     OBJECT IDENTIFIER ::= { rs232Conformance 2 }

-- compliance statements

rs232Compliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "The compliance statement for SNMPv2 entities
        which have RS-232-like hardware interfaces."

    MODULE -- this module
        MANDATORY-GROUPS { rs232Group }

        GROUP rs232AsyncGroup
        DESCRIPTION
            "The Async group is mandatory only for those
            SNMPv2 entities which have asynchronous
            interfaces Rs-232-like."

        GROUP rs232SyncGroup
        DESCRIPTION
            "The Synch group is mandatory only for those
            SNMPv2 entities which have synchronous
            interfaces Rs-232-like."
    ::= { rs232Compliances 1 }
```

-- units of conformance

```
rs232Group      OBJECT-GROUP
  OBJECTS { rs232Number, rs232PortIndex, rs232PortType,
            rs232PortInSigNumber, rs232PortOutSigNumber,
            rs232PortInSpeed, rs232PortOutSpeed,
            rs232PortInFlowType, rs232PortOutFlowType,
            rs232InSigPortIndex, rs232InSigName,
            rs232InSigState, rs232InSigChanges,
            rs232OutSigPortIndex, rs232OutSigName,
            rs232OutSigState, rs232OutSigChanges }
  STATUS      current
  DESCRIPTION
    "A collection of objects providing information
     applicable to all RS-232-like interfaces."
  ::= { rs232Groups 1 }

rs232AsyncGroup OBJECT-GROUP
  OBJECTS { rs232AsyncPortIndex, rs232AsyncPortBits,
            rs232AsyncPortStopBits, rs232AsyncPortParity,
            rs232AsyncPortAutobaud, rs232AsyncPortParityErrs,
            rs232AsyncPortFramingErrs, rs232AsyncPortOverrunErrs }
  STATUS      current
  DESCRIPTION
    "A collection of objects providing information
     applicable to asynchronous RS-232-like interfaces."
  ::= { rs232Groups 2 }

rs232SyncGroup OBJECT-GROUP
  OBJECTS { rs232SyncPortIndex, rs232SyncPortClockSource,
            rs232SyncPortFrameCheckErrs,
            rs232SyncPortTransmitUnderrunErrs,
            rs232SyncPortReceiveOverrunErrs,
            rs232SyncPortInterruptedFrames,
            rs232SyncPortAbortedFrames }
  STATUS      current
  DESCRIPTION
    "A collection of objects providing information
     applicable to synchronous RS-232-like interfaces."
  ::= { rs232Groups 3 }

rs232SyncSDLCGroup OBJECT-GROUP
  OBJECTS { rs232SyncPortRole,
            rs232SyncPortEncoding,
            rs232SyncPortRTSControl,
            rs232SyncPortRTSCTSDelay,
            rs232SyncPortMode,
            rs232SyncPortIdlePattern,
```

```
        rs232SyncPortMinFlags }
STATUS   current
DESCRIPTION
    "A collection of objects providing information
    applicable to synchronous RS-232-like interfaces
    running SDLC."
 ::= { rs232Groups 4 }
```

END

5. Acknowledgements

This memo was produced by the IETF Character MIB Working Group.

6. References

- [1] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1442, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.
- [2] McCloghrie, K., and M. Rose, Editors, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II", STD 17, RFC 1213, Hughes LAN Systems, Performance Systems International, March 1991.
- [3] Galvin, J., and K. McCloghrie, "Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1445, Trusted Information Systems, Hughes LAN Systems, April 1993.
- [4] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1448, SNMP Research, Inc., Hughes LAN Systems, Dover Beach Consulting, Inc., Carnegie Mellon University, April 1993.
- [5] McCloghrie, K., and F. Kastenholz, "Evolution of the Interfaces Group of MIB-II", RFC 1573, Hughes LAN Systems, FTP Software, January 1994.
- [6] Stewart, B., "Definitions of Managed Objects for Character Stream Devices using SMIV2", RFC 1658, Xyplex, Inc., July 1994.
- [7] Kastenholz, F., "The Definitions of Managed Objects for the Link Control Protocol of the Point-to-Point Protocol", RFC 1471, FTP Software, Inc., June 1993.

7. Security Considerations

Security issues are not discussed in this memo.

8. Author's Address

Bob Stewart
Xyplex, Inc.
295 Foster Street
Littleton, MA 01460

Phone: 508-952-4816
Fax: 508-952-4887
EMail: rlstewart@eng.xyplex.com

