```
-- HIPPI Switch MIB v1.01
-- Last updated April 14, 1997
-- Contact Mark Doppke
--   Essential Communications Inc.
--   e-mail: marck@esscom.com

--##########################################################################
--hippiSwitchMIB                   OBJECT IDENTIFIER ::= { ecExperimental 1 }
--hippiSwitchMIB                   OBJECT IDENTIFIER ::= { ecExperimental 3 }
--   switchObjs                    OBJECT IDENTIFIER ::= { hippiSwitchMIB 1 }
--      backPlaneTable             OBJECT-TYPE       ::= { switchObjs 1 }
--         backPlaneEntry          OBJECT-TYPE       ::= { backPlaneTable 1}
--            backPlaneIndex       OBJECT-TYPE       ::= { backPlaneEntry 1}
--            backPlaneNumber      OBJECT-TYPE       ::= { backPlaneEntry 2}
--            backPlaneCard        OBJECT-TYPE       ::= { backPlaneEntry 3}
--      switch                     OBJECT IDENTIFIER ::= { switchObjs2 2 }
--         switchDescription       OBJECT-TYPE       ::= { switch 1 }
--         switchNumOfPorts        OBJECT-TYPE       ::= { switch 2 }
--      scc                        OBJECT IDENTIFIER ::= { switchObjs 3 }
--         sccDescription          OBJECT-TYPE       ::= { scc 1 }
--         sccDateTime             OBJECT-TYPE       ::= { scc 2 }
--         sccAdminStatus          OBJECT-TYPE       ::= { scc 3 }
--         sccOperStatus           OBJECT-TYPE       ::= { scc 4 }
--         sourceRouteTable        OBJECT-TYPE       ::= { scc 5 }
--         sourceRouteEntry        OBJECT-TYPE       ::= { sourceRouteTable 1
}
--            inPortIndex          OBJECT-TYPE       ::= { sourceRouteEntry 1
}
--            outPortIndex         OBJECT-TYPE       ::= { sourceRouteEntry 2
}
--            routeStatus          OBJECT-TYPE       ::= { sourceRouteEntry 3
}
--         huntGroupTable          OBJECT-TYPE       ::= { scc 6 }
--            huntGroupEntry       OBJECT-TYPE       ::= { huntGroupTable 1 }
--               huntGroupIndex    OBJECT-TYPE       ::= { huntGroupEntry 1 }
--               huntGroupOutPortList OBJECT-TYPE    ::= { huntGroupEntry 2 }
--         huntGroupOrderTable     OBJECT-TYPE       ::= { scc 7 }
--            huntGroupOrderEntry  OBJECT-TYPE       ::= { huntGroupOrderTable
1 }
--               huntGroupOrderIndex OBJECT-TYPE     ::= { huntGroupOrderEntry
1 }
--               huntGroupOrderList OBJECT-TYPE      ::= { huntGroupOrderEntry
2 }
--         destRouteTable          OBJECT-TYPE       ::= { scc 8 }
--            destRouteEntry       OBJECT-TYPE       ::= { destRouteTable 1 }
--               destRouteIndex    OBJECT-TYPE       ::= { destRouteEntry 1 }
--               inputPortIndex    OBJECT-TYPE       ::= { destRouteEntry 2 }
--               huntGroup         OBJECT-TYPE       ::= { destRouteEntry 3 }
--   switchEvents                  OBJECT IDENTIFIER ::= { hippiSwitchMIB 2 }
--      switchEventsV2             OBJECT IDENTIFIER ::= { switchEvents 0 }
--         switchHeapBad           NOTIFICATION-TYPE ::= { switchEventsV2 1 }
--   switchConformance             OBJECT IDENTIFIER ::= { hippiSwitchMIB 3 }
--      switchGroups               OBJECT IDENTIFIER ::= { switchConformance
1}
--      switchCompliance           OBJECT IDENTIFIER ::= { switchConformance
2}
--##########################################################################
ESSENTIAL-COMMUNICATIONS-HIPPI-SWITCH DEFINITIONS ::= BEGIN

IMPORTS
      MODULE-IDENTITY, OBJECT-TYPE,
      NOTIFICATION-TYPE,Unsigned32,enterprises
            FROM SNMPv2-SMI
      DisplayString
            FROM SNMPv2-TC
      ecExperimental
            FROM ESSENTIAL-COMMUNICATIONS-GLOBAL-REG;
```

```
hippiSwitchMIBMod MODULE-IDENTITY
       LAST-UPDATED "9704140000Z"
       ORGANIZATION "Essential Communications Inc."
       CONTACT-INFO "Marck Doppke
                                 email marck@esscom.com"
       DESCRIPTION         "Design attemp at a standard HIPPI switch MIB v1.01"
       ::= { ecExperimental 3 }

--Main branches of the MIB.
essentialCommunications OBJECT IDENTIFIER ::= { enterprises 2159 }
ecRoot            OBJECT IDENTIFIER ::= { essentialCommunications 1 }
ecExperimental    OBJECT IDENTIFIER ::= { ecRoot 6 }
hippiSwitchMIB    OBJECT IDENTIFIER ::= { ecExperimental 1 }
switchObjs        OBJECT IDENTIFIER ::= { hippiSwitchMIB 1 }
switchEvents      OBJECT IDENTIFIER ::= { hippiSwitchMIB 2 }
switchConformance OBJECT IDENTIFIER ::= { hippiSwitchMIB 3 }
switchGroups      OBJECT IDENTIFIER ::= { switchConformance 1 }
switchCompliance  OBJECT IDENTIFIER ::= { switchConformance 2 }

--#####################################################################
--First branch of this MIB containing the hardware objects.
--#####################################################################

--#####################################################################
--The backplane hardware description.
--#####################################################################
backPlaneTable      OBJECT-TYPE
       SYNTAX       SEQUENCE OF BackPlaneEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
                            "This table represent all of the slots in a HIPPI
switch.
                            None of the rows can be added to or deleted by the
user."
       ::= { switchObjs 1 }

backPlaneEntry OBJECT-TYPE
       SYNTAX              BackPlaneEntry
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
                            "A row in the table describing one slot in the
switch back
                            plane. These rows can not be added or deleted by the
user."
       INDEX        { backPlaneIndex } --Identifies a row in the slot table.
       ::= { backPlaneTable 1 }

BackPlaneEntry ::= SEQUENCE
{
       backPlaneIndex            Unsigned32,
       backPlaneNumber           Unsigned32,
       backPlaneCard             INTEGER
}

backPlaneIndex OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   not-accessible
       STATUS       current
       DESCRIPTION
                            "The table index for this slot on the backplane."
       ::= { backPlaneEntry 1 }

backPlaneNumber     OBJECT-TYPE
       SYNTAX       Unsigned32
```

```
        MAX-ACCESS   read-only
        STATUS             current
        DESCRIPTION  "The slot number as seen printed on the switch
                           (backPlaneIndex + 1)"
        ::= { backPlaneEntry 2 }

backPlaneCard OBJECT-TYPE
        SYNTAX       INTEGER { none(1), serial(2), parrallel(3) }
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION  "The type of card present in this slot of the backplane on
                           the switch"
        ::= { backPlaneEntry 3 }

--The Switch hardware description.
switch OBJECT IDENTIFIER ::= { switchObjs 2 }

switchDescription OBJECT-TYPE
        SYNTAX       DisplayString
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION "Returns the description, vendor, and version numbers of
the switch."
        ::= { switch 1 }

switchNumOfPorts OBJECT-TYPE
        SYNTAX       Unsigned32
        MAX-ACCESS   read-only
        STATUS       current
        DESCRIPTION  "The number of slots in this switch. (Max number of media
                           access cards."
        ::= { switch 2 }

--The Switch Control Card (SCC) hardware description.
scc OBJECT IDENTIFIER ::= { switchObjs 3 }
sccDescription OBJECT-TYPE
        SYNTAX       DisplayString
        MAX-ACCESS   read-only
        STATUS             current
        DESCRIPTION  "The model, vendor, and version number of the switch
control
                           card."
        ::= { scc 1 }

sccDateTime OBJECT-TYPE
        SYNTAX       DisplayString
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION  "The time/date in the real time clock. Format is yymmdd
hhmmss"
        ::= { scc 2 }

sccAdminStatus OBJECT-TYPE
        SYNTAX       INTEGER { normal(0),reset(1) }
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION "The desired state of the switch."
        ::= { scc 3 }

sccOperStatus OBJECT-TYPE
        SYNTAX       INTEGER { normal(0),reset(1) }
        MAX-ACCESS   read-write
        STATUS       current
        DESCRIPTION "The current state of the switch."
        ::= { scc 4 }

--SRC_ROUTES
```

```
--A table that contains the enable/disable information for each input port-
--output port pair.
sourceRouteTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF SourceRouteEntry
       MAX-ACCESS   not-accessible
       STATUS             current
       DESCRIPTION "This table holds all of the source routes for this
switch."
       ::= { scc 5 }

sourceRouteEntry OBJECT-TYPE
       SYNTAX       SourceRouteEntry
       MAX-ACCESS   not-accessible
       STATUS             current
       DESCRIPTION "A row in the source route table."
       INDEX        { inPortIndex, outPortIndex }
       ::= { sourceRouteTable 1 }

SourceRouteEntry ::= SEQUENCE
{
       inPortIndex        Unsigned32,  --read-only.
       outPortIndex Unsigned32,  --read-only.
       routeStatus        Unsigned32   --read-write
}

inPortIndex OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The input port number of the source route that is being
                            modified."
       ::= { sourceRouteEntry 1 }

outPortIndex OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The output port number of the source route that is being
                            modified."
       ::= { sourceRouteEntry 2 }

routeStatus OBJECT-TYPE
       SYNTAX       Unsigned32 ( 0..1 )
       MAX-ACCESS   read-create
       STATUS       current
       DESCRIPTION "The enabled/disabled status of this input/output port
pair."
       ::= { sourceRouteEntry 3 }

--##########################################################################
--Hunt Group Table (Output port sets).
--The huntgroup table defines which output ports are in each huntgroup set.
--##########################################################################
huntGroupTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF HuntGroupEntry
       MAX-ACCESS   not-accessible
       STATUS             current
       DESCRIPTION "This table holds all of the huntgroups for this  switch."
       ::= { scc 6 }

huntGroupEntry OBJECT-TYPE
       SYNTAX       HuntGroupEntry
       MAX-ACCESS   not-accessible
       STATUS             current
       DESCRIPTION "A row in the huntgroup table."
       INDEX        { huntGroupIndex }
       ::= { huntGroupTable 1 }
```

```
HuntGroupEntry ::= SEQUENCE
{
       huntGroupIndex                   Unsigned32,
       huntGroupOutPortList      DisplayString (1..80)
}

huntGroupIndex OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The huntgroup number."
       ::= { huntGroupEntry 1 }

huntGroupOutPortList OBJECT-TYPE
       SYNTAX       DisplayString
       MAX-ACCESS   read-create
       STATUS       current
       DESCRIPTION "A text string containing the list of all of the output
ports
                           that are in the this huntgroup. It is the agent's
                           responsiblility to parse and varify the correctness
of the
                           values. It is recommended that the agent support the
following
                           string formats.
                                  val = portNumber | portRange [,| val]
                                  portNumber = 1..MaxNumberOfPorts
                                  portRange  = portNumber-portNumber.
                           EXAMPLE '1 2,4-6 7-10'"
       ::= { huntGroupEntry 2 }

--##########################################################################
--Hunt Group Order. A table that contains that the huntgroups are processed
--in when more that 1 huntgroup contends for the same output port.
--table 16x32. f(outPort) = hgList.
--##########################################################################
huntGroupOrderTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF HuntGroupOrderEntry
       MAX-ACCESS   not-accessible
       STATUS              current
       DESCRIPTION "This table holds all of the huntgroup order info
                           for this  switch."
       ::= { scc 7 }

huntGroupOrderEntry OBJECT-TYPE
       SYNTAX       HuntGroupOrderEntry
       MAX-ACCESS   not-accessible
       STATUS              current
       DESCRIPTION "The order hunt groups are processed in when more than one
                           hunt group contends for the same output port."
       INDEX        { huntGroupIndex }
       ::= { huntGroupOrderTable 1 }

HuntGroupOrderEntry ::= SEQUENCE
{
       huntGroupOrderIndex       Unsigned32,
       huntGroupOrderList        DisplayString (1..80)
}

huntGroupOrderIndex OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The slot number for the output port."
       ::= { huntGroupOrderEntry 1 }
```

```
huntGroupOrderList OBJECT-TYPE
       SYNTAX       DisplayString
       MAX-ACCESS   read-create
       STATUS       current
       DESCRIPTION "A text string containing the order of all the hunt groupts
                            that use this port. The first entries in the list
get the
                            output port first if more than one huntgroup is
contending
                            for the port. It is the agent's
                            responsiblility to parse and varify the correctness
of the
                            values. It is recommended that the agent support the
following
                            string formats.
                                   val = hgNumber | hgRange [,| val]
                                   hgNumber = 1..MaxNumberOfHuntGroups
                                   hgRange  = hgNumber-hgNumber.
                            EXAMPLE '1 2,4-6 7-10'"
       ::= { huntGroupOrderEntry 2 }

--#######################################################################
--The destination routing table contains the huntgroup number
--(if any) for each defined logical address and input port pair.
--tabel max is 4kx16. f(la,inport) = HG
--#######################################################################
destRouteTable OBJECT-TYPE
       SYNTAX       SEQUENCE OF DestRouteEntry
       MAX-ACCESS   not-accessible
       STATUS              current
       DESCRIPTION "This table holds all of destination routes (logical
                            addressing)  for this  switch."
       ::= { scc 8 }

destRouteEntry OBJECT-TYPE
       SYNTAX       DestRouteEntry
       MAX-ACCESS   not-accessible
       STATUS              current
       DESCRIPTION "A table containing destination routes."
       INDEX        { destRouteIndex, inputPortIndex }
       ::= { destRouteTable 1 }

DestRouteEntry ::= SEQUENCE
{
       destRouteIndex            Unsigned32,
       inputPortIndex            Unsigned32,
       huntGroup                 Unsigned32
}

destRouteIndex OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The logical address of this route."
       ::= { destRouteEntry 1 }

inputPortIndex OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   read-only
       STATUS       current
       DESCRIPTION "The input port that this destination route is defined
for."
       ::= { destRouteEntry 2 }

huntGroup OBJECT-TYPE
       SYNTAX       Unsigned32
       MAX-ACCESS   read-create
```

```
       STATUS       current
       DESCRIPTION "The huntgroup for this logical address input port
pairing."
       ::= { destRouteEntry 3 }


--#####################################################################
--The second branch of this MIB that contains the events.
--#####################################################################
--This is here to easily translate SNMPv1 traps.
switchEventsV2       OBJECT IDENTIFIER ::= { switchEvents 0 }

--Sample event
switchHeapBad       NOTIFICATION-TYPE
--  OBJECTS { }
       STATUS       current
       DESCRIPTION "The memory heap in the SCC EEMM is corrupted."
       ::= { switchEvents 1 }

--#####################################################################
--The third branch of this MIB that contains the conformance information
--in the compliance and group branches. Wait till the MIB is done.
--#####################################################################

END
```