Event Monitor

– this section is intended to be read in conjunction with the Introduction

Television Systems Limited.

Vanwall Road, Maidenhead, Berkshire, SL6 4UB

Telephone +44 (0)1628 676200, FAX +44 (0)1628 676299

Event Monitor

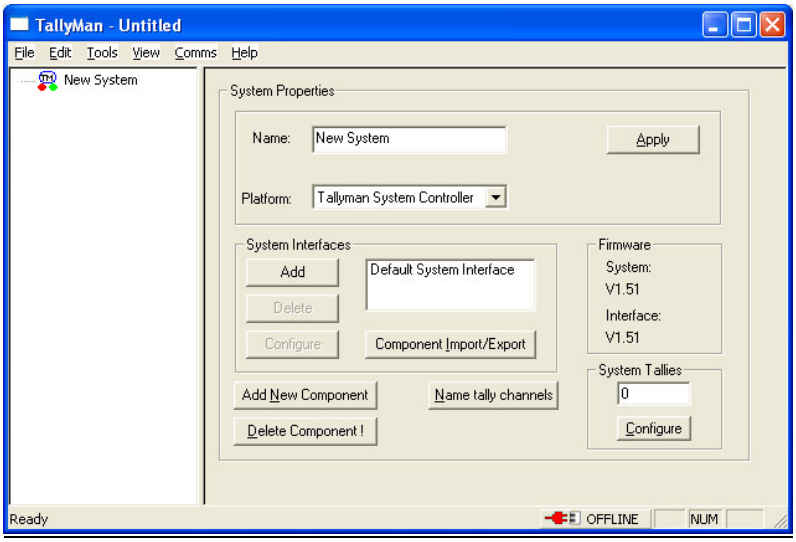
These enhanced features are only available from TallyMan Version 1.51 (December 2006).

1.0 Introduction

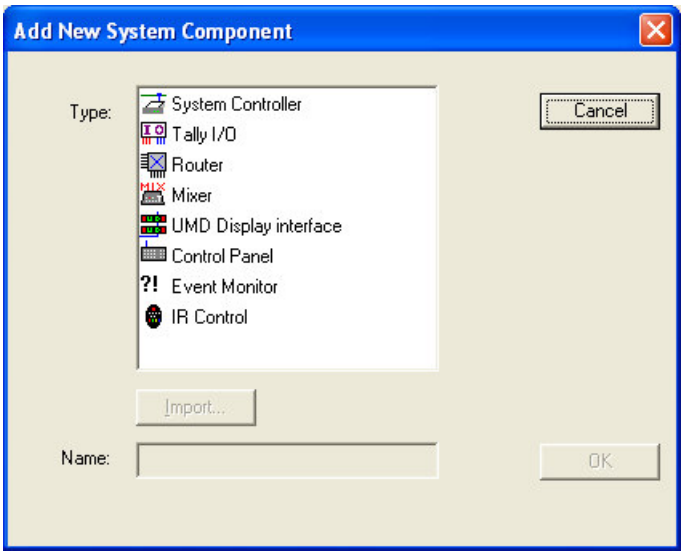
This feature allows enhanced control of tallies and routers. In addition it is possible to upload a new TallyMan configuration file on receipt of an incoming trigger (tally/GPI/router change).

2.0 Adding an Event

Click on **Add New Component** on the main screen.



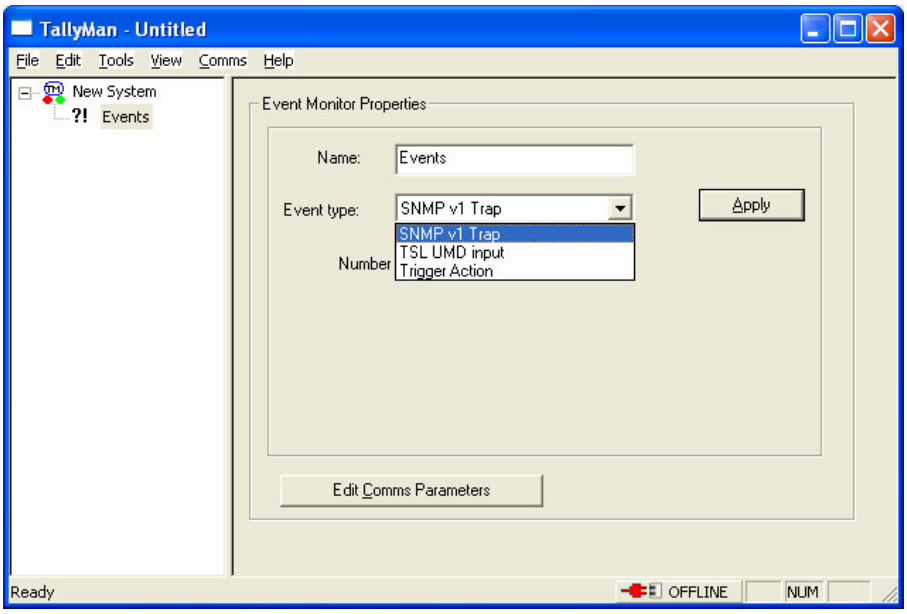
From the **Add New Component** list select **Event Monitor**.



When a new component is added to the system it must be given a Name before the OK button becomes active.

This Name will be seen in the system tree on the left side of the screen.

Selecting the Event Monitor type



Select the **Event** type from the drop down list.

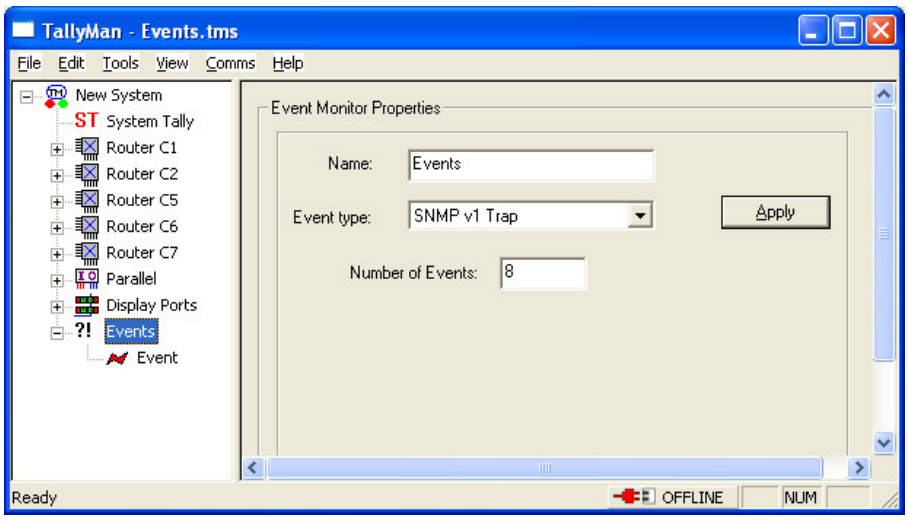
3.0 SNMP Trap - only implemented on the product TMC-1

This description is intended as an overview for this facility which is currently (Nov 2007) only implemented on the TMC-1 product.

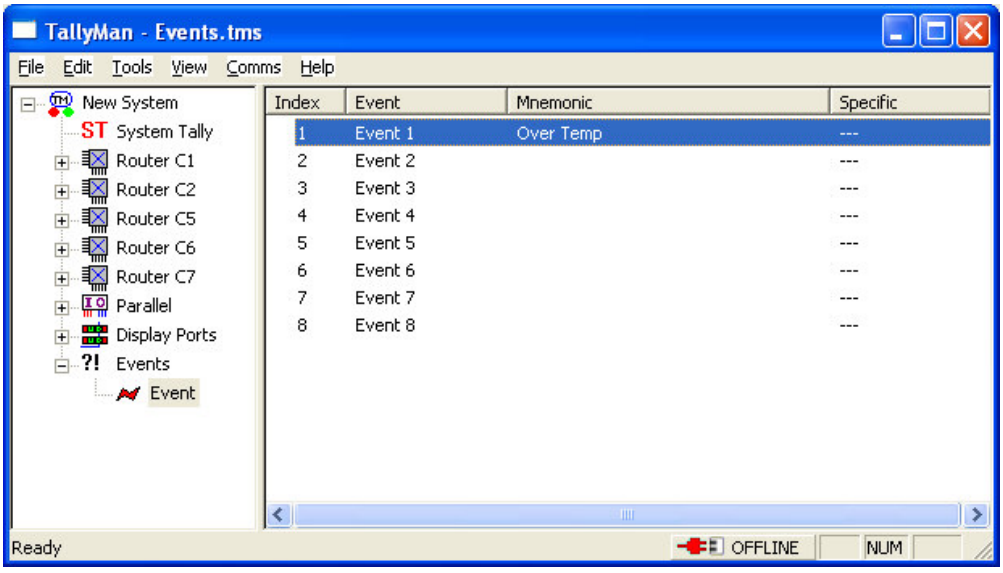
Simple Network Management Protocol

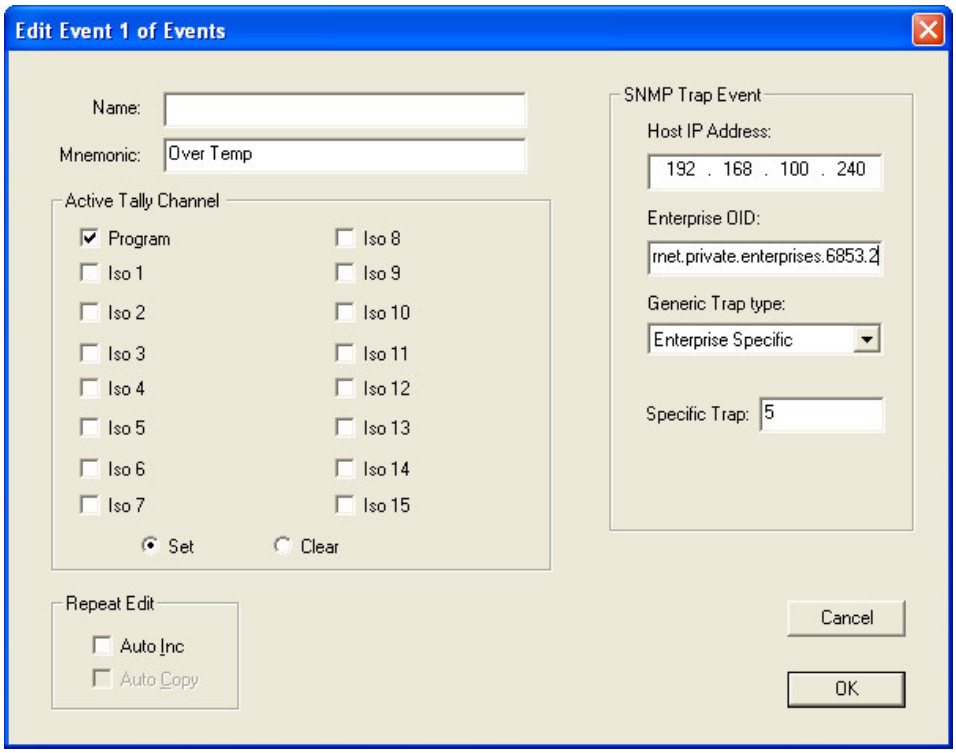
This is a communication protocol between management stations and managed objects, (such as routers and MDUs) and makes use of Management Information Bases (MIB) .
SNMP uses a specified set of commands and queries. An MIB will contain information on these commands and on the target objects i.e. controllable entities or potential sources of status information.

A SNMP Event is shown as added to an existing system. Eight events have been added in this example.



Click on **Event** to open the list.





The Host IP address has been entered. This is the IP address of the unit that has to be monitored.

The Enterprise OID (Object IDentifiers) has to be entered; for TSL units this is 6853. The string: **.iso.org.dod.internet.private.enterprises**. (.1.3.6.1.4.1) is added automatically (presumed).

The string 6853 is unique to TSL.

If, say, a router is being monitored, the 6853.2 number shown above will be different, using the published numbers for that manufacturer's router. The 2 in this example would be the TSL MIB number.

The number currently published for TSL is actually only 6853

From the MIB: `tslMIB OBJECT IDENTIFIER ::= { enterprises 6853 }`

The generic trap type is shown as **Enterprise Specific** from the drop down list.

The **Specific Trap** will be the number that corresponds to the information required, as described in the MIB (Management Information Base). See Appendix 1 for the TSL MDU example.

Decide whether the Active Tally Channel should be **Set** or **Clear** for the tally/mnemonic.

Notes:

IANA, Internet Assigned Numbers Authority, assigns the IP and OID numbers.

http://www.iana.org/assignments/enterprise-numbers

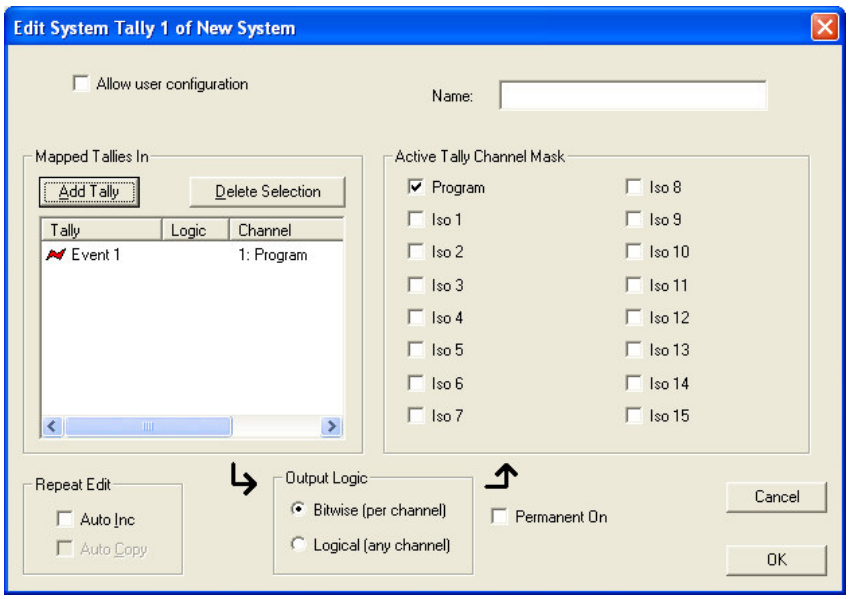
and: http://www.alvestrand.no/objectid/1.3.6.1.4.html

The Firewall will need to be disabled (the factory default on a TMC-1) or the Ports 161/162 specifically opened.

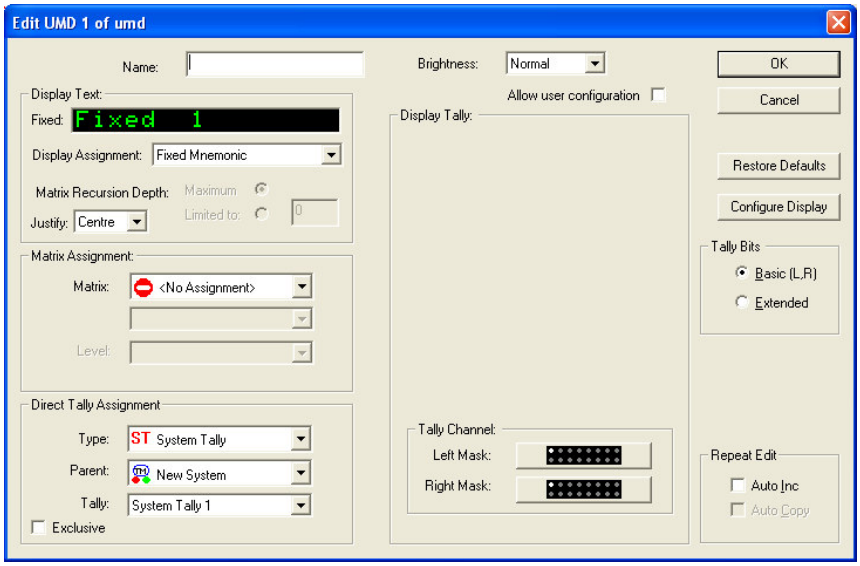
When that Trap is sent, the mnemonic entered (**Over Temp** this example) will be shown on a UMD.

The UMD screen is shown below.

- Add the event to the UMD via a **System Tally**.



The use of a **System Tally** is essential for operation and also allow other **Events** to be added to the UMD.

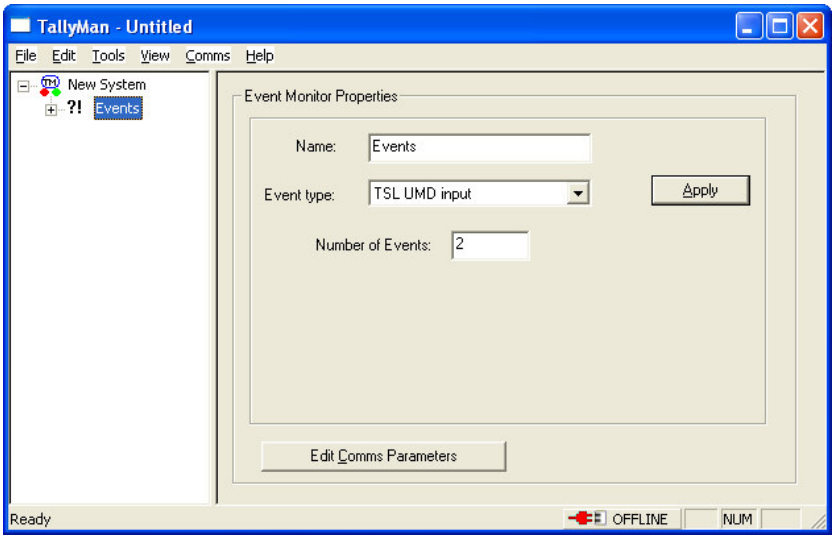


This is a very specific use of the UMD. The text and Tally Lights will change in the event of a trap being received.

- Write the file to the TMC-1 and go on-line.

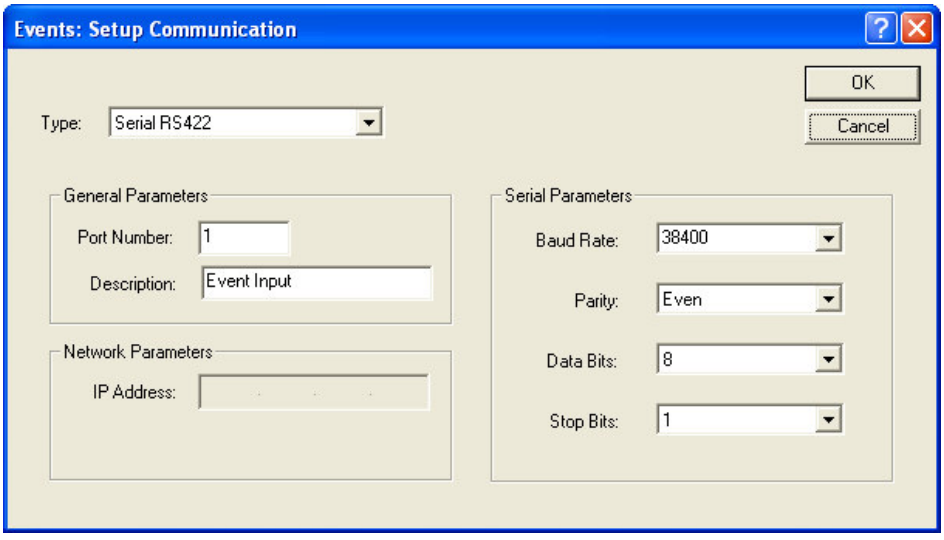
Correct communications will show by a green dot against the **Events** entry when on-line, as is normal.

4.0 TSL UMD Input

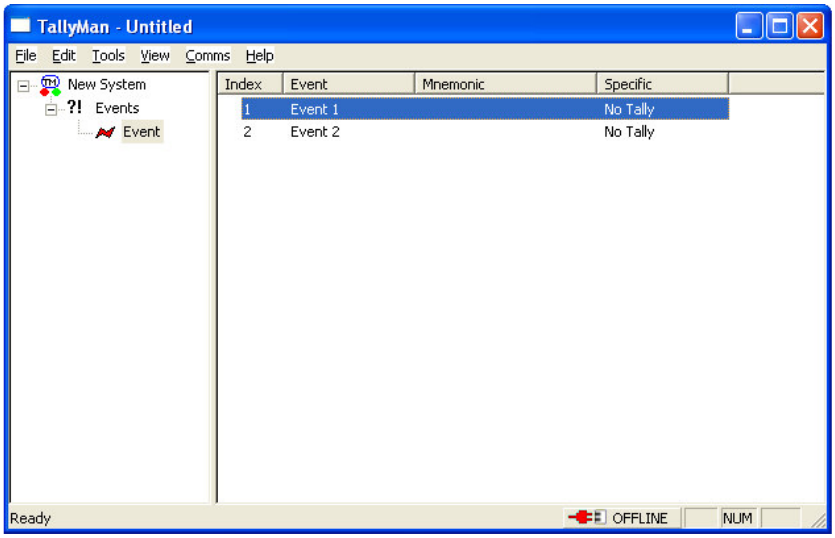


Edit Comms Parameters for the data incoming port into the TallyMan unit. The TallyMan controller will expect standard TSL UMD Protocol.

Incoming tallies start at Address 000. Two are shown above so they will be at address 000 and 001. These may then be mapped to control a tally lamp on any UMD or may be output to the Tally Output pins etc.

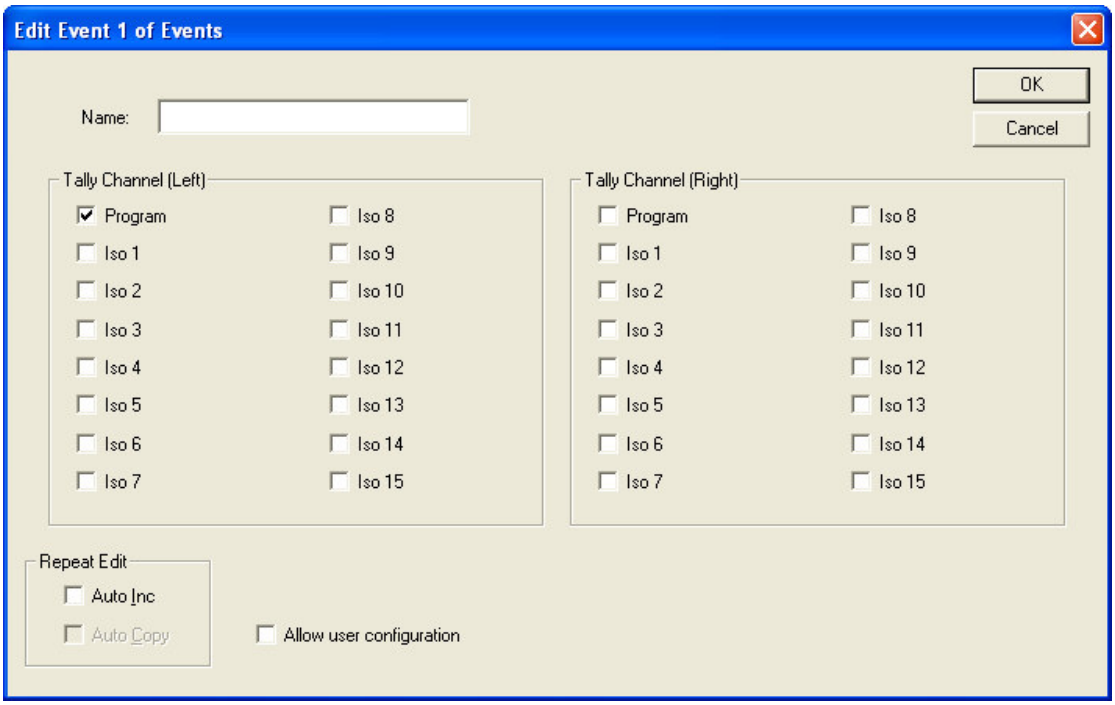


Select the Event.

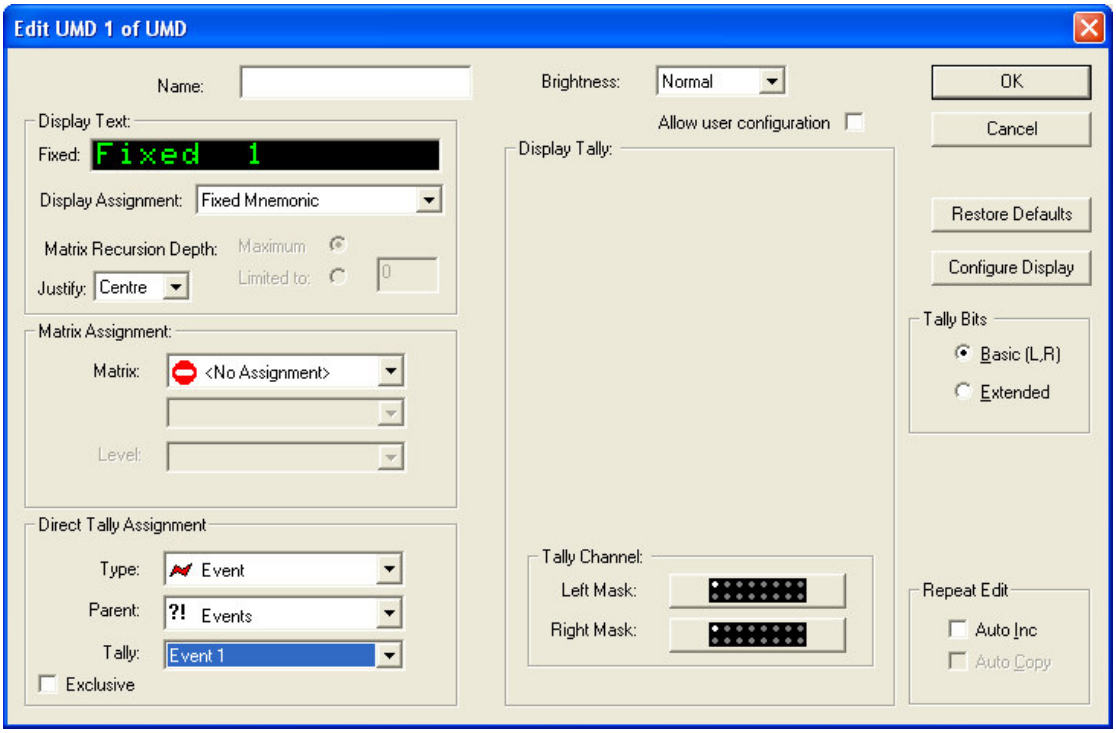


An incoming tally will show its status in the Specific column.

The incoming tally signal may be set to any Channel. Please see the Tally Section for more information.

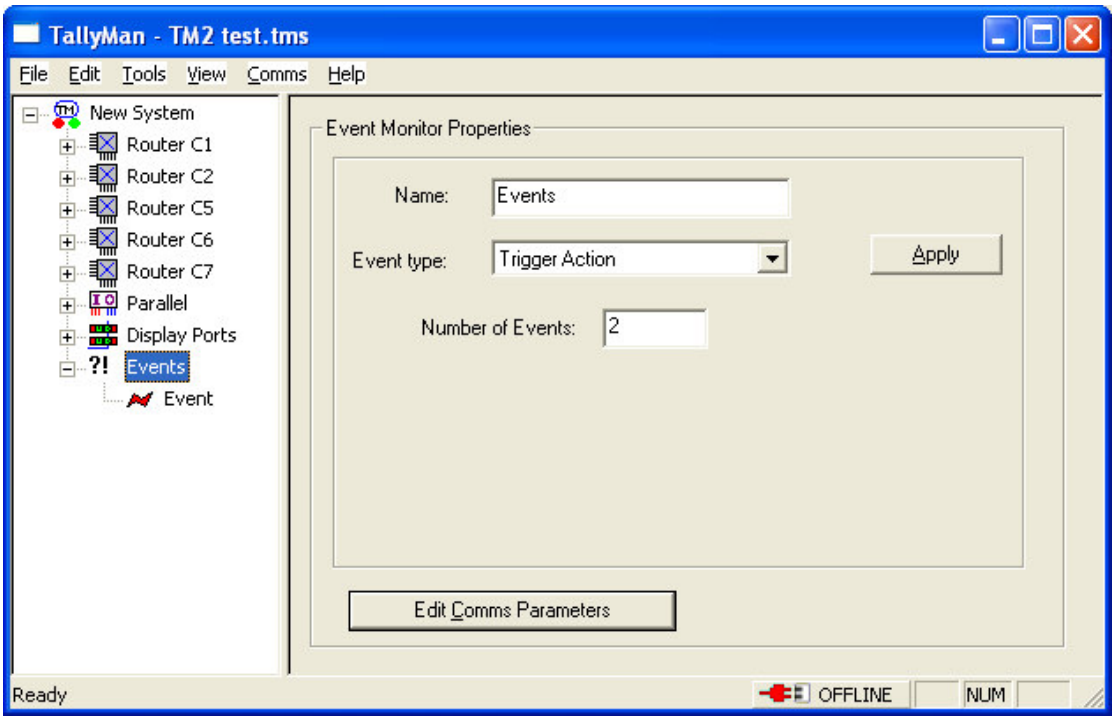


This may then be mapped to a tally function such as activating a tally LED on a UMD.

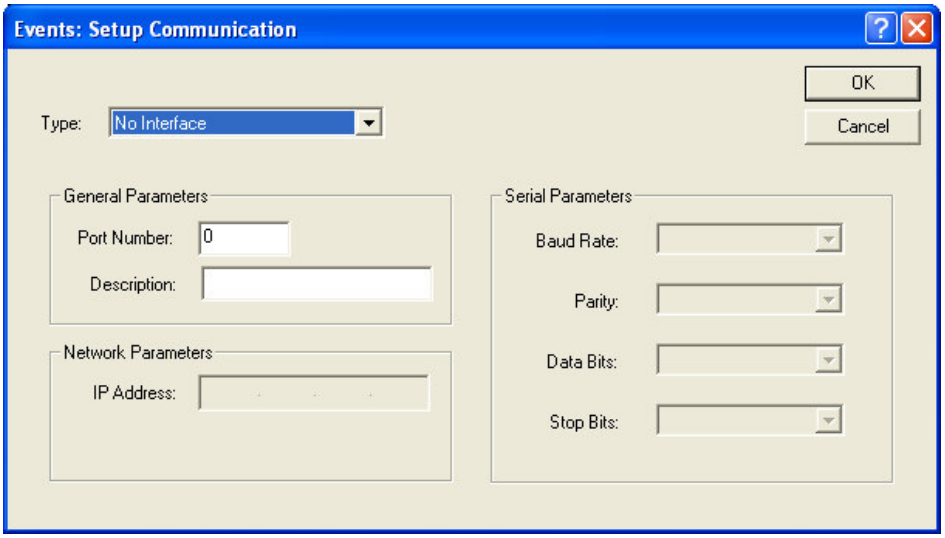


Note: The correct Tally Channel mask must be set in the usual way.

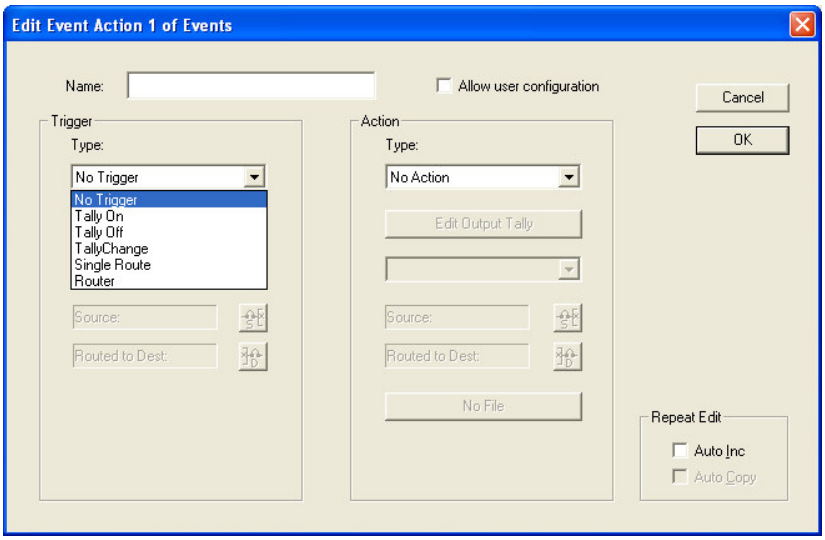
5.0 Trigger Action



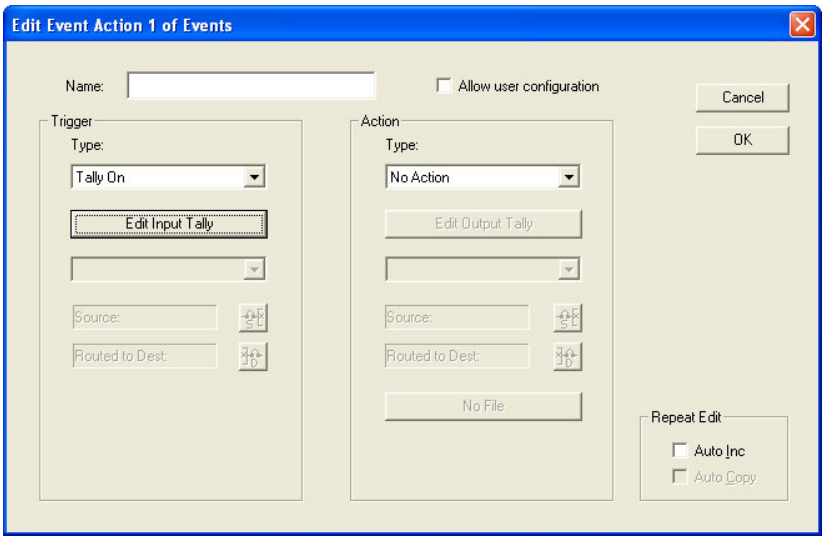
There is no interface or port to be set with this event.



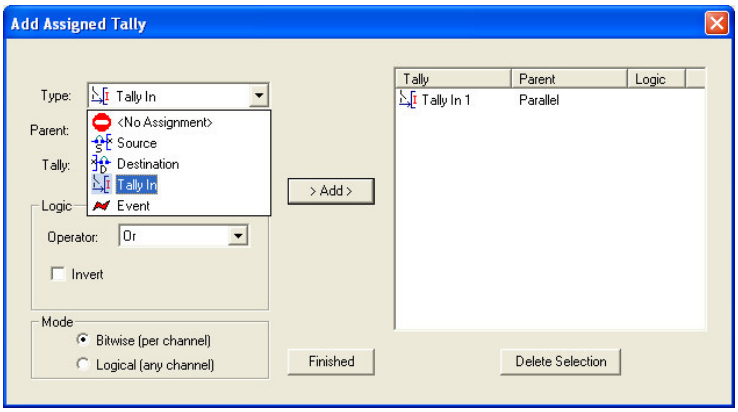
Select the Event.

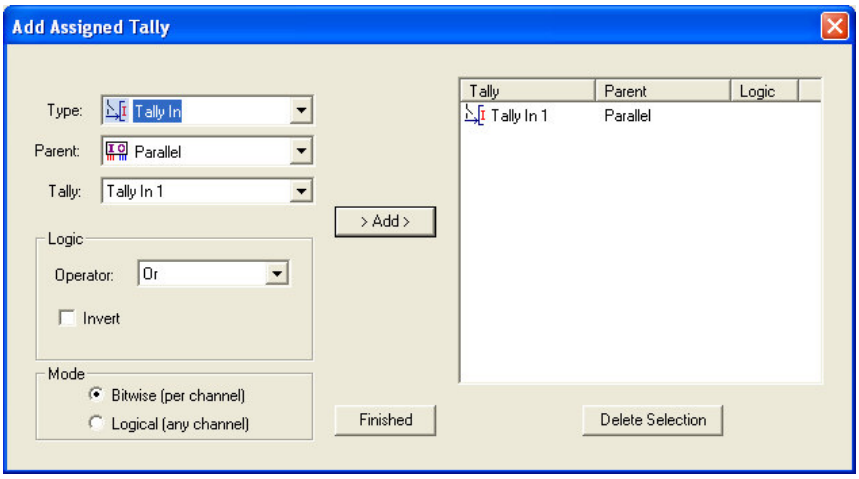


Say we are going to use a Tally (or incoming GPI) to trigger an event.

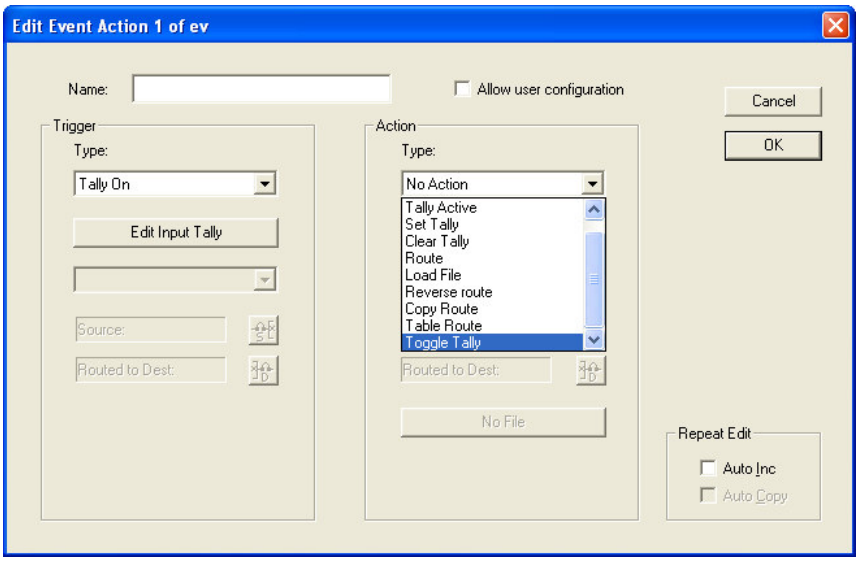


Select the required input tally from the available list.



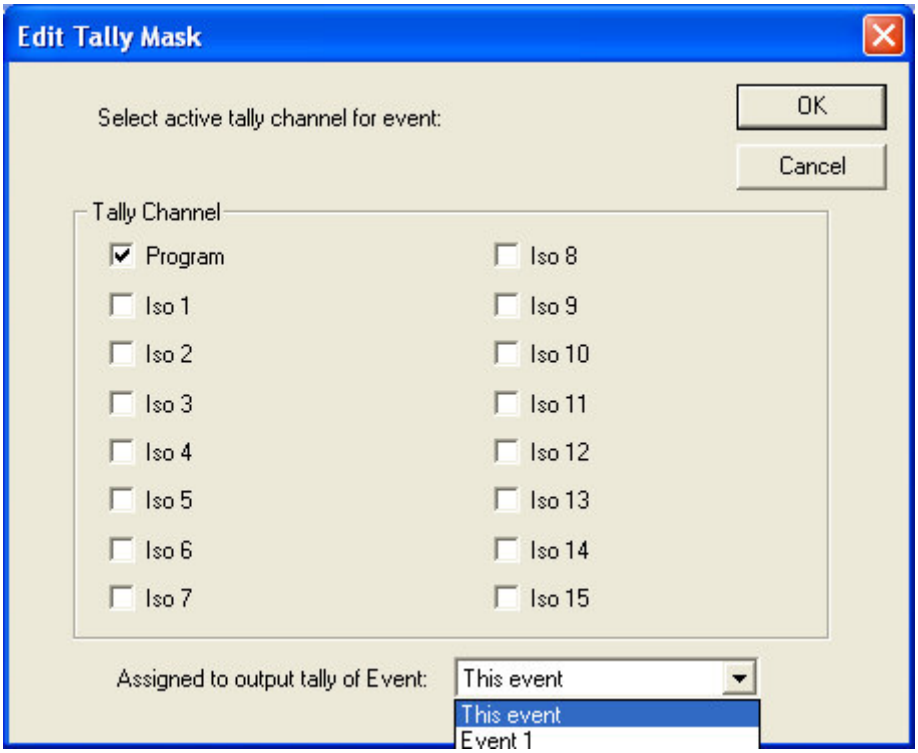


Now select the required **Action**.



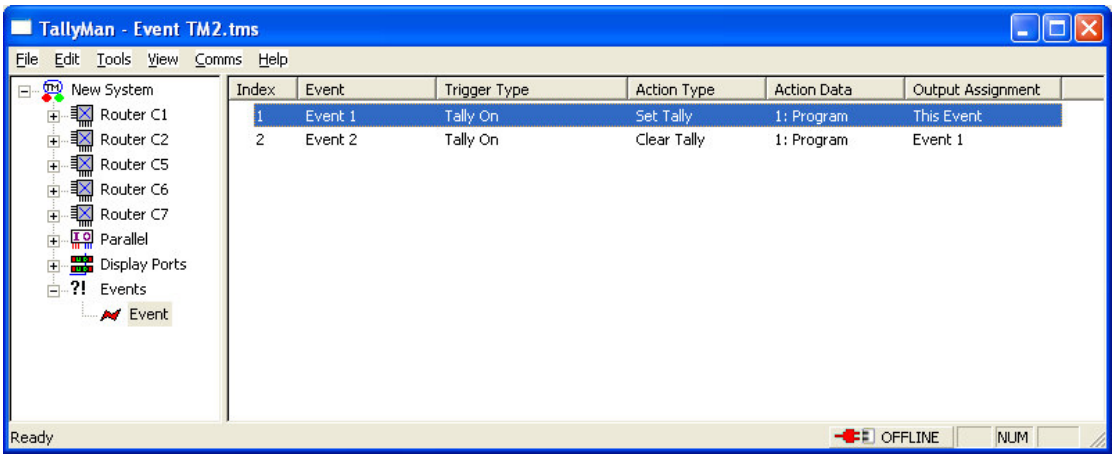
No Action:	Nothing will happen.
Tally Active:	The tally will follow the Trigger status, i.e. ON/OFF.
Set Tally:	Tally is turned ON (remains ON even when the trigger has gone).
Clear Tally:	Tally is turned OFF (remains OFF even when the trigger has gone).
Route:	A specific router crosspoint will be made.
Load File:	A Router file (.rtr) or a TallyMan File (.tms) will be loaded (Care!).
Reverse route:	Inputs/Outputs are reversed on the controlled router.
Copy Route:	Routers are effectively paralleled for crosspoint control.
Table Route:	This allow control whereby when one destination is switched, another destination is switched as a slave. This may be on the same or on a different router.
Toggle Tally:	With an incoming tally, say, the tally output will toggle on and then off with the next action.

Tally Active/Set Tally/Clear Tally

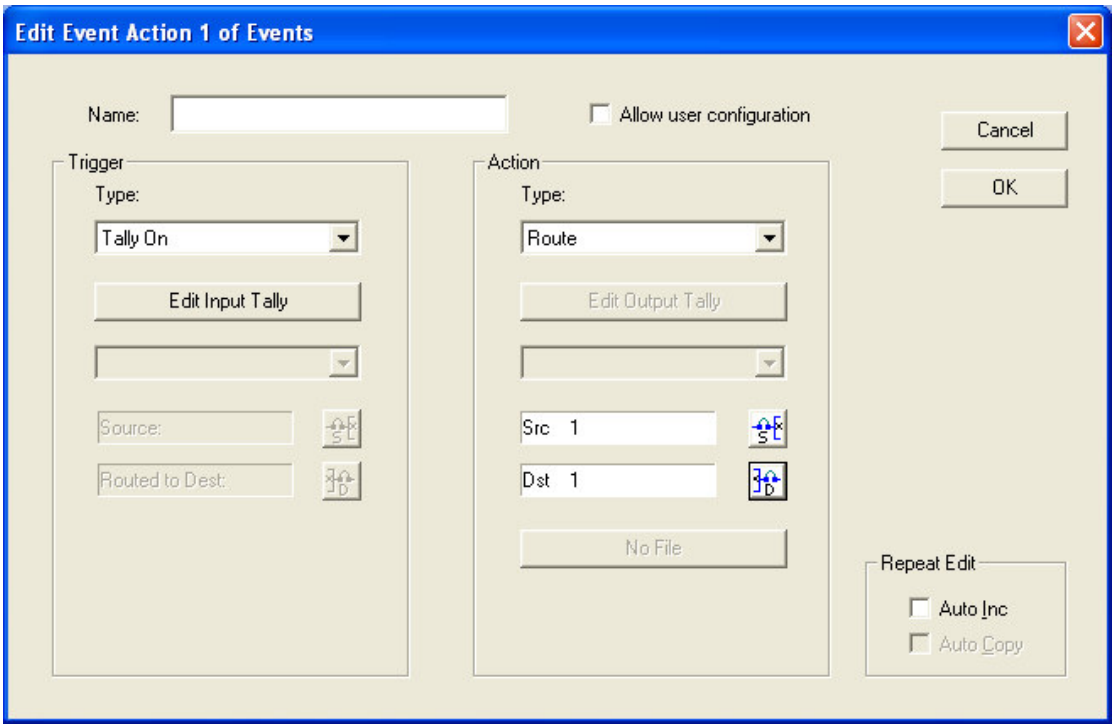


The **Assigned to output tally of Event** is an option for the Events configured.

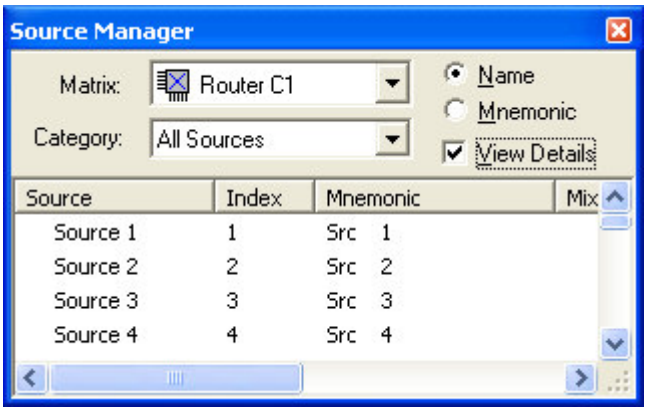
You may, for example, set a tally with Event 1 and turn this tally off in Event 1 using action from Event 2.



You may use a Tally ON for example to make a router crosspoint.



Clicking the buttons will call up the router lists.

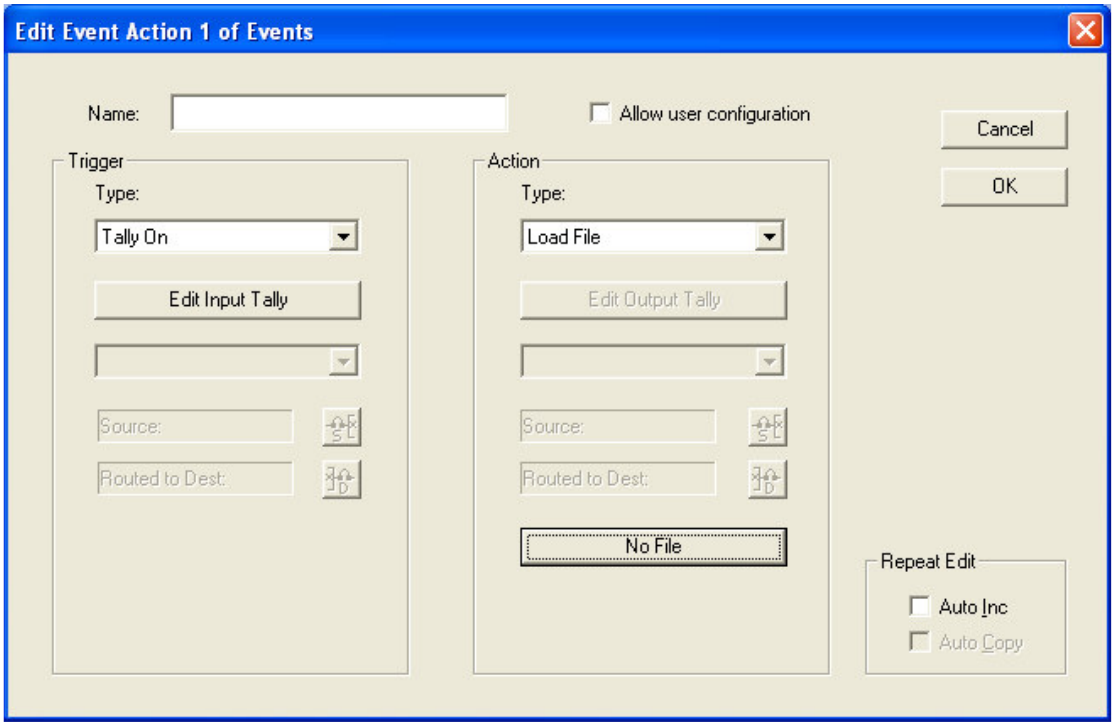


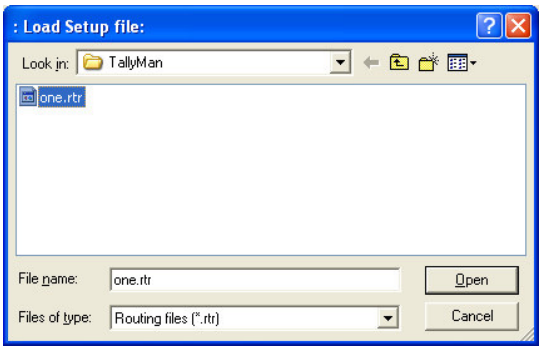
Double click on the required Source or Destination.

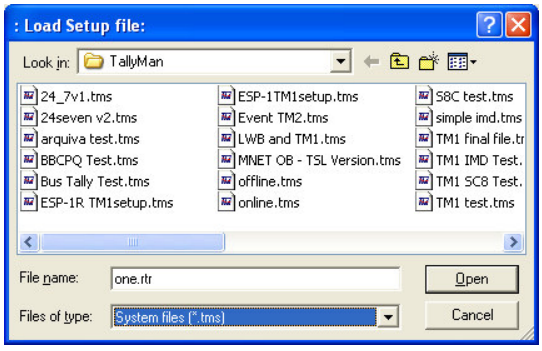
Load File.

This will load the router **rtr** status file . or a new .**tms** Config file.

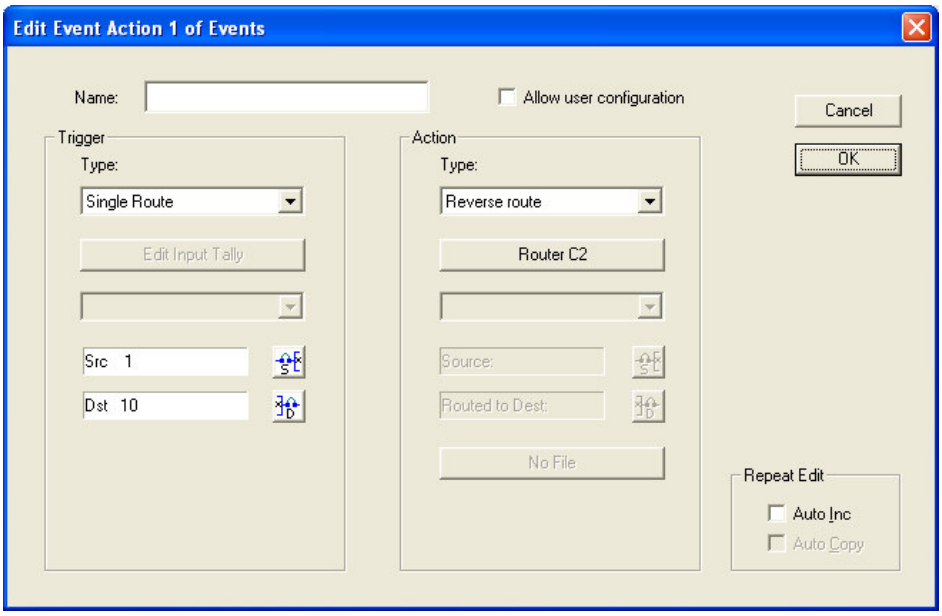
Note: With the .**tms** file you need to be sure that you have programmed the new file with the appropriate module in order to return to the original file if you wish to retain this automation.



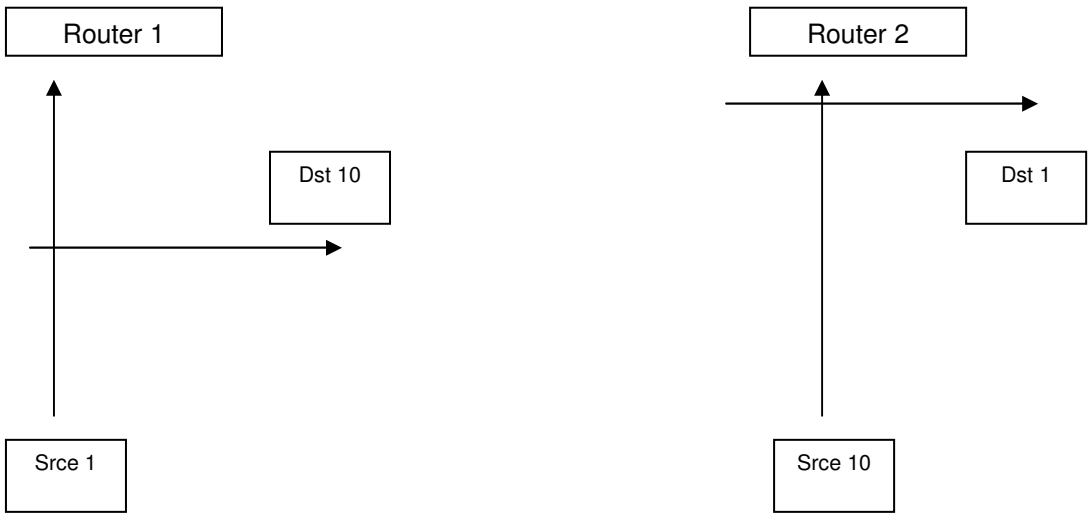


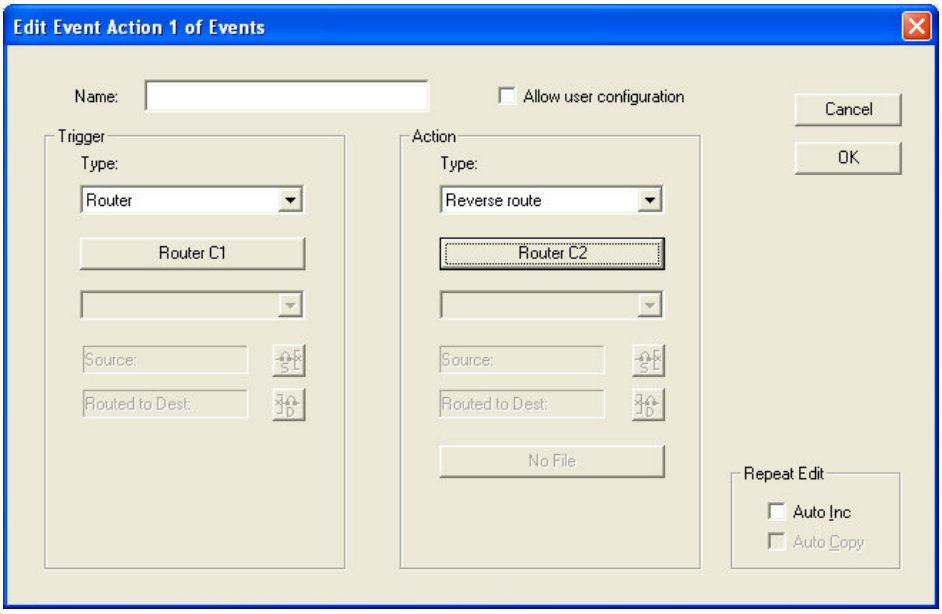


Reverse Route



In the case shown, when Router 1 has Source 1 routed to Destination 10, Router 2 will have Source 10 routed to Destination 1.





This will reverse route the second Router C2 completely.

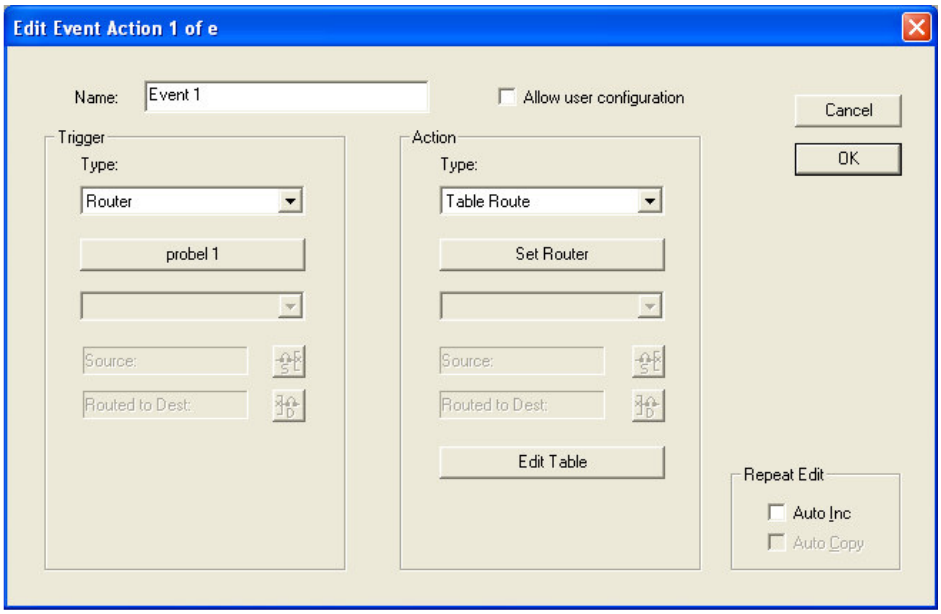
Copy Route

This effectively parallels router control.

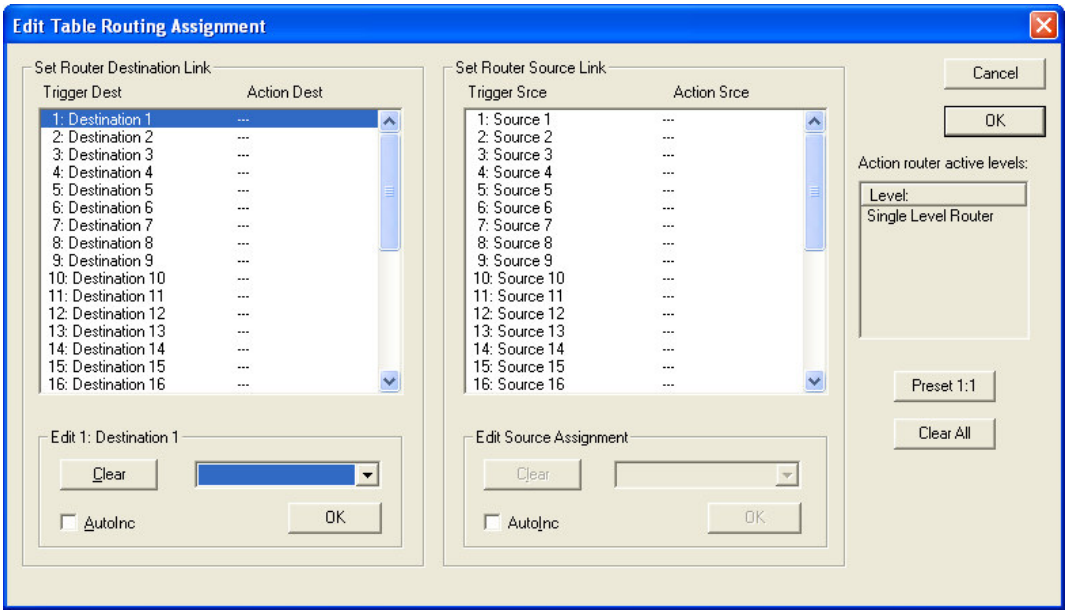
Router 1: Source 1 is routed to Dest 10
Router 2: Source 1 is routed to Dest 10.

Table Route

This allow control whereby when one destination is switched, another destination is switched as a slave. This may be on the same or on a different router.

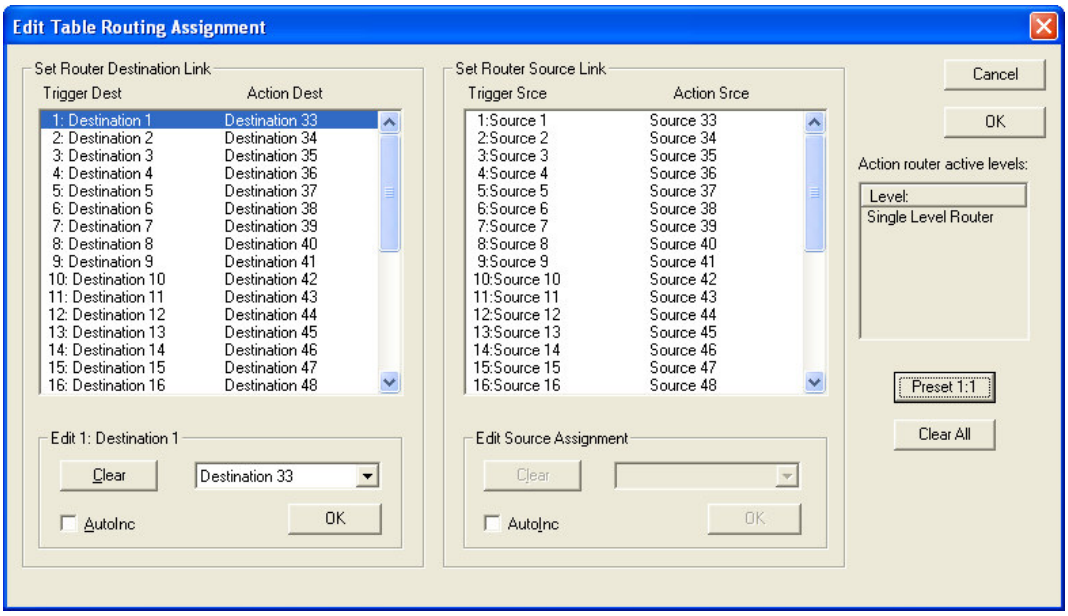


- Select Edit Table



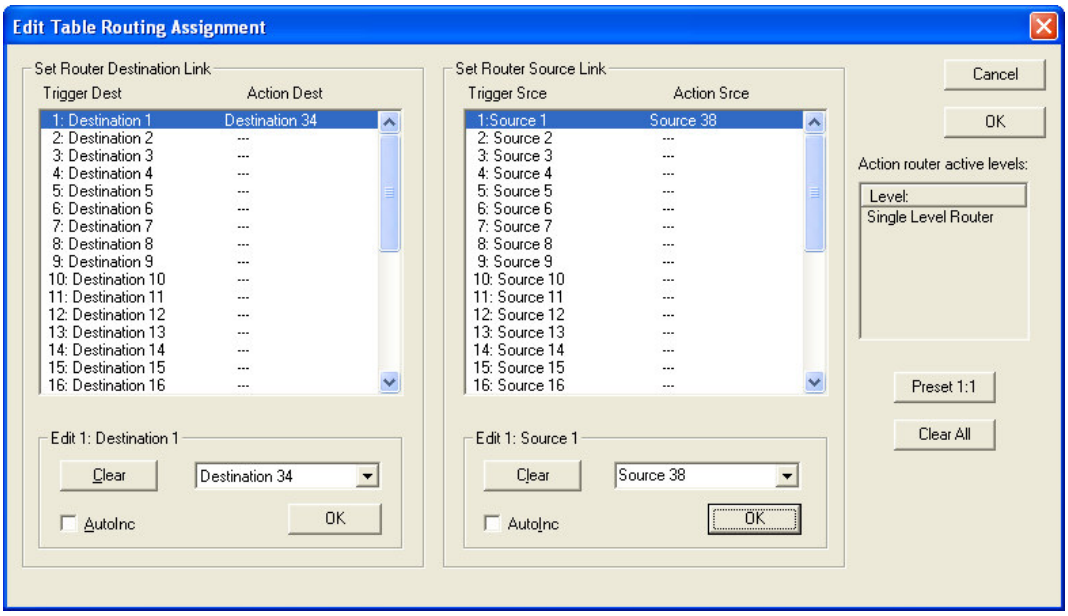
Preset 1: 1 will match each destination and sources with their counterpart or individual destinations to other destinations etc. may be mapped.

1:1 setting selected.



The two routers are shown. One starts with the system destination number as Destination 1 and the second router is shown with the router destination set as Destination 33.

Individual Destinations to destinations selected.



Appendix 1

```
TSL-MIB DEFINITIONS ::= BEGIN

IMPORTS
enterprises, Opaque
FROM RFC1155-SMI
OBJECT-TYPE
FROM RFC-1212
TRAP-TYPE
FROM RFC-1215;

-- MODULE-IDENTITY
-- FROM SNMPv2-SMI;

-- TSL_MIB; SNMP v1 agent definitions.

-- the following only allowed in SMIv2 (also 0 enumeration of
integers)

-- As of 08/08/03, includes enterprise specific trap definitions
(RFC1215)

-- tslMIB MODULE-IDENTITY
--     LAST-UPDATED "0308080000Z"
--     ORGANIZATION "Television Systems Ltd"
--     CONTACT-INFO "
--     Tim Whittaker
--     Television Systems Ltd
--     Unit 4, King's Grove
--     Maidenhead
--     Berkshire
--     SL6 4DP
--
--     Tel + 44 1628 687200
--     Email: timw@televisionsystems.ltd.uk"
--     DESCRIPTION "MIB module for all TSL products"
--     ::= { enterprises 6853 }

DisplayString ::= OCTET STRING


--  SMIv1 definition of module

tslMIB OBJECT IDENTIFIER ::= { enterprises 6853 }


------------------ Winsoft specific MIB

-- DELETED for mdu12 hardware

-------------------------------------------------

-------------- generic alarm MIB (all TSL equipment capable of SNMP
alarms)

alarm OBJECT IDENTIFIER ::= { tslMIB 2 }

alarmIdent OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-only
```

```
    STATUS mandatory
    DESCRIPTION
    "Equipment alarms description and version"
    ::= { alarm 1 }

------- alarm table

alarmTable OBJECT-TYPE
    SYNTAX SEQUENCE OF AlarmEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
    "The table of alarm entries"
    ::= { alarm 2 }

alarmEntry OBJECT-TYPE
    SYNTAX AlarmEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
    " An alarm entry in the table"
      INDEX { alarmTableIndex }
    ::= { alarmTable 1}


AlarmEntry ::= SEQUENCE
      {
            alarmTableIndex INTEGER,
            alarmType INTEGER,
            alarmIndex INTEGER,
            alarmText DisplayString,
            alarmState INTEGER,
            alarmPolarity INTEGER,
            alarmData Opaque
      }

alarmTableIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "The index into the table"
    ::= { alarmEntry 1 }

alarmType OBJECT-TYPE
    SYNTAX INTEGER
    {
            internal(1),      -- general internal to equipment alarm
            gpi(2),                 -- from external GPI,
alarmPolarity determines alarmState
            outputFail(3),    -- eg MDU12 output fuse, etc
            psuFail(4)        -- alarmData is text describing failure
(eg rail values etc)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "Alarm type"

    ::= { alarmEntry 2 }
```

```
alarmIndex OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "Alarm type number"
    ::= { alarmEntry 3 }

alarmText OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
    "Alarm description"
    ::= { alarmEntry 4 }

alarmState OBJECT-TYPE
    SYNTAX INTEGER
    {
            inactive(1),
        active(2)
    }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "Alarm state"
    ::= { alarmEntry 5 }

alarmPolarity OBJECT-TYPE
    SYNTAX INTEGER
    {
            notApplicable(1),
        normallyOpen(2),
        normallyClosed(3)
    }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
    "Alarm active polarity (notApplicable for non-gpi alarms)"
    ::= { alarmEntry 6 }


alarmData OBJECT-TYPE
    SYNTAX Opaque
    ACCESS read-only
    STATUS optional
    DESCRIPTION
    "Additional alarm data of variable length, according to alarm
type."
    ::= { alarmEntry 7 }


------- end of table

alarmTotal OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "The total number of alarms in the table"
    ::= { alarm 3 }
```

```
alarmLocation OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
    "The physical location of the equipment generating the alarm"
    ::= { alarm 4 }


alarmEqptTemp OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-only
    STATUS optional
    DESCRIPTION
    "Equipment temperature (in degrees Centigrade)"
    ::= { alarm 5 }

alarmEqptTempHi OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-write
    STATUS optional
    DESCRIPTION
    "Equipment temperature alarm point (degrees Centigrade)"
    ::= { alarm 6 }

alarmTrap TRAP-TYPE
      ENTERPRISE  tslMIB
      VARIABLES
      {
            alarmTableIndex,
            alarmType,
            alarmIndex,
            alarmText,
            alarmState,
            alarmPolarity,
            alarmData
      }
      DESCRIPTION
      "An entry in the alarm table has changed state"
      ::= 4

alarmEqptTempHiTrap TRAP-TYPE
      ENTERPRISE  tslMIB
      VARIABLES
      {
            alarmEqptTemp
      }
      DESCRIPTION
      "The equipment temperature has exceeded the maximum allowed"
      ::= 5

alarmEqptTempOkTrap TRAP-TYPE
      ENTERPRISE  tslMIB
      VARIABLES
      {
            alarmEqptTemp
      }
      DESCRIPTION
      "The equipment temperature is now within limits"
      ::= 6
--------------------------------------------------
```

```
-------------- MDU12 specific MIB

mdu12 OBJECT IDENTIFIER ::= { tslMIB 3 }

mdu12Ident OBJECT-TYPE
    SYNTAX DisplayString
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "Equipment description and version"
    ::= { mdu12 1 }


mduPowerOn OBJECT-TYPE
    SYNTAX INTEGER
      {
            simultaneous(1),
            sequential(2),
            delayed(3)
      }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
    "MDU power-on output sequence"
    ::= { mdu12 2 }

mduSeqDelay OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
    "Sequential mode delay between outputs"
    ::= { mdu12 3 }


mduOutputTable OBJECT-TYPE
    SYNTAX SEQUENCE OF MduOutputEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
    "Table of output controls"
    ::= { mdu12 4 }

mduOutputEntry OBJECT-TYPE
    SYNTAX MduOutputEntry
    ACCESS not-accessible
    STATUS mandatory
    DESCRIPTION
    " An entry in the output table"
    INDEX {mduOutputIndex}
    ::= { mduOutputTable 1}

MduOutputEntry ::= SEQUENCE
      {
        mduOutputIndex INTEGER,
        mduOutputState  INTEGER,
        mduOutputDelay  INTEGER
      }

mduOutputIndex OBJECT-TYPE
```

```
    SYNTAX INTEGER(1..12)
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "Output number"
    ::= { mduOutputEntry 1 }

mduOutputState OBJECT-TYPE
    SYNTAX INTEGER
      {
            off(1),
            on(2),
            locked-Off(3),    -- locked by admin web page, cannot
change via SNMP
            locked-On(4)      -- locked by admin web page, cannot
change via SNMP
      }
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
    "MDU Output status"
    ::= { mduOutputEntry 2 }

mduOutputDelay OBJECT-TYPE
    SYNTAX INTEGER
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION
    "Output on delay from power on (Delay mode only)"
    ::= { mduOutputEntry 3 }

-- End of table

mduPowerStatus OBJECT-TYPE
    SYNTAX INTEGER
    {
            totalLoss(1),
            input1OK(2),
            input2OK(3),
            allOk(4)          -- note: a single input MDU would
report allOk if power is present
      }
    ACCESS read-only
    STATUS mandatory
    DESCRIPTION
    "Status of MDU power inlets"
    ::= { mdu12 5 }


mduPowerStatusTrap TRAP-TYPE
      ENTERPRISE  tslMIB
      VARIABLES
      {
            mduPowerStatus
      }
      DESCRIPTION
      "The power input to the MDU has changed state"
      ::= 7

------------
END
```