CHAPTER 5

MIB Commands

Use the MIB commands described in this chapter to display and modify the values of the MIB objects that are used to control a LightStream 2020 multiservice ATM switch (LS2020 switch). (See the chapter entitled "LightStream 2020 MIB Reference" for additional information about MIB objects.) The MIB commands are as follows:

browse	Browse the MIB tree.
getsnmp	Display the value of a MIB object.
getnextsnmp	Display the value of the object in the MIB tree that follows the specified object.
setsnmp	Change the state of the specified MIB object.
walksnmp	Display the values of all MIB objects in the MIB tree starting with the specified object.

browse

Use the **browse** command to walk through the MIB tree and display the value of an object in the MIB tree.

Syntax

```
browse [mib-address]
```

Arguments

Use the optional *mib-address* argument to specify the point in the MIB tree at which the browse display starts. If you do not specify any argument, the display starts at the top of the MIB tree (at the iso object).

The address can be specified as a path of dot-separated numbers, as a variable name, or as a combination (where only the first element is a variable name). Thus, 1.3.6.1.2.1, mgmt.1, and mib all refer to the same variable. (See the chapter entitled "LightStream 2020 MIB Reference" for a description of MIB addresses.)

Examples

The following example shows how to use the **browse** command to display objects in the `mgmt` subtree:

```
cli> browse mgmt

mgmt:
       1) mib Enter line number to go down, 'u' to go up, 'q' or 'e' to quit browse.
browse>
```

Because there is only one object under the mgmt object, you select it by entering the menu number **1**:

```
browse> 1

mgmt.mib:
        1) system
        2) interfaces
        3) at
        4) ip
        5) icmp
        6) tcp
        7) udp
        8) egp
        9) transmission
        10) snmp
        11) dot1dBridge
        12) rmon
Enter line number to go down, 'u' to go up, 'q' or 'e' to quit browse.
browse>
```

There are 12 objects under the mib object. To select the object system, enter the menu number **1:**

```
browse> 1

mgmt.mib.system:
        1) sysDescr
        2) sysObjectID
        3) sysUpTime
        4) sysContact
        5) sysName
        6) sysLocation
        7) sysServices
Enter line number to go down, 'u' to go up, 'q' or 'e' to quit browse.
browse>
```

All of the objects under the system object are "leaves" with no further branches under them. If you select one of those objects from the menu, the program displays its value and returns to the mgmt.mib.system menu. In the following example, sysContact is the object chosen (menu item 4):

```
browse> 4
-----------------------------------------------------
Name: sysContact.0            Value: Lisa Bloch

mgmt.mib.system:
        1) sysDescr
        2) sysObjectID
        3) sysUpTime
        4) sysContact
        5) sysName
        6) sysLocation
        7) sysServices Enter line number to go down, 'u' to go up, 'q' or 'e' to quit
browse.
browse>
```

To explore a different branch of the `mib` subtree, you can enter **u** to go up a level and return to the mib object:

```
browse> u

mgmt.mib:
        1) system
        2) interfaces
        3) at
        4) ip
        5) icmp
        6) tcp
        7) udp
        8) egp
        9) transmission
        10) snmp
        11) dot1dBridge
        12) rmon Enter line number to go down, 'u' to go up, 'q' or 'e' to quit browse.
browse>
```

To exit to the `cli>` prompt, enter **q** (or **e**):

```
browse> q
Leaving browse
cli>
```

getsnmp

Use the **getsnmp** command to display the value of a specified MIB object. Given the addresses of one or more MIB objects, **getsnmp** displays the value of the MIB object at each address.

Syntax

```
getsnmp mib-address [mib-address [ ... mib-address]]
```

Arguments

The *mib-address* argument specifies the address of a MIB object that you want to display. As an optional, repeated argument, it specifies the address of an additional MIB object or objects that you want to display.

The address of a MIB object has two dot-separated parts:

- The last part is a numeric qualifier, the object ID. The object ID for an isolated leaf is 0. In a table, the object ID depends upon how the table entries are indexed. For example, 4002 represents card 4, port 2. The object ID can contain more than one index.
- The first part of the address can be a path of dot-separated numbers, a MIB object name, or a combination (where only the first element is a MIB object name). Thus, sysDescr.0, system.1.0, mib.1.1.0, and 1.3.6.1.2.1.1.1.0 all refer to the same MIB object. See the chapter entitled "LightStream 2020 MIB Reference" for more information about MIB addresses.

To display the value of more than one object, specify more than one object identifier. You can use the **walksnmp** command to identify MIB objects and their addresses.

Note Any *mib-address* arguments after the first are optional.

Examples

The following examples illustrate different kinds of MIB addresses. The first example shows how to use the **getsnmp** command, specifying the leaf name (plus the object identifier 0) for each object:

```
cli> getsnmp sysContact.0 sysServices.0
Name: sysContact.0             Value: Lisa Bloch
Name: sysServices.0            Value: 78
cli>
```

The following example shows use of two other kinds of MIB addresses for the same pair of objects, a complete path of numbers from the top of the MIB, and a path beginning with the mgmt object:

```
cli> getsnmp 1.3.6.1.2.1.1.4.0 mgmt.1.1.7.0
Name: sysContact.0             Value: Lisa Bloch
Name: sysServices.0            Value: 78
cli>
```

getnextsnmp

Use the **getnextsnmp** command to display the value of the object that comes after the specified object in the MIB tree.

If you specify the last variable (“leaf” object) in a subtree, the command displays the first variable in the next subtree.

Syntax

```
getnextsnmp mib-address [mib-address [ ... mib-address]]
```

Arguments

The *mib-address* argument specifies the address of the MIB object that is just before the object that you want to display. As an optional, repeated argument, the MIB-address argument specifies the address of an additional MIB object whose next-following neighbor in the MIB tree you want to display.

The address can be a path of dot-separated numbers, a variable name, or a combination of the two, where only the first element is a variable name. Thus, 1.3.6.1.2.1.1.1, mib.1.1, system.1, and sysDescr all refer to the same variable. See the description of the **getsnmp** command for additional information about MIB addresses, and see the chapter entitled “LightStream 2020 MIB Reference” for a detailed description.

To display the values of more than one object, specify more than one object identifier. You can use the **walksnmp** command to identify MIB objects and their addresses.

Note Any *mib-address* arguments after the first are optional.

Examples

The following examples illustrate different kinds of MIB addresses. The first example shows how to use the **getnextsnmp** command, specifying the leaf name (plus the object identifier 0) for each object:

```
cli> getnextsnmp sysContact.0 sysServices.0
Name: sysName.0                Value: Comet
Name: ifNumber.0               Value: 10007
cli>
```

The following example shows use of two other kinds of MIB addresses for the same pair of objects, a complete path of numbers from the top of the MIB, and a path beginning with the mgmt object:

```
getnextsnmp 1.3.6.1.2.1.1.4.0 mgmt.1.1.7.0
Name: sysName.0                Value: lstb7
Name: ifNumber.0               Value: 10007
cli>
```

setsnmp

Use the **setsnmp** command to change the value of the specified MIB object.

Note The **setsnmp** command does not verify the values of its arguments. If there is a CLI **set** command for the parameter that you want to set, use it instead of the **setsnmp** command.

The **setsnmp** command requires protected mode. See the **protected** command in the chapter entitled "CLI Control Commands".

The **setsnmp** command requires that the read/write community name attribute be set first to a name that has been assigned the value `write` in the mma.communities file. If the read/write community name has not been set first, the **setsnmp** command fails, because the default community name "public" is read only. See the description of the command **set snmp community** in the chapter entitled "The Set Command" and see the *LightStream 2020 Network Operations Guide* for information on setting the read/write community name attribute.

Caution Do not manipulate MIB objects unless you are familiar with SNMP.

Syntax

```
setsnmp MIBaddress value
```

Arguments

MIBaddress Identifies the MIB variable to be set.

The address can be a path of dot-separated numbers, a variable name, or a combination of the two, where only the first element is a variable name. Thus, 1.3.6.1.2.1.1.1.0, mib.1.1.0, system.1.0, and sysDescr.0 all refer to the same variable. See the preceding description of the **getsnmp** command for additional information about MIB addresses, and see the chapter entitled "LightStream 2020 MIB Reference" for a detailed description. You can use the **walksnmp** command, described in the following section, to identify MIB objects and their addresses.

value Specifies the new value for the MIB object at *MIBaddress*. If the *value* argument contains spaces, you must use quotation marks around it. The value may be a number, a string, or an object ID preceded by a colon, in the form **:***MIBaddress2*. To determine the appropriate argument type for the specified MIB object, type **?** after the first *MIBaddress* argument.

Example

The following example shows use of **setsnmp** to change the name of the contact person for the system (note quotation marks):

```
*cli> setsnmp sysContact.0 "Tom Smith"
Name: sysContact.0              Value: Tom Smith
*cli>
```

The following example shows how to type **?** after a partial command to display online help for that command. In the example, **?** is typed after a **setsnmp** command to determine what type of value can be assigned to the specified object, ifDescr.1000 (the interface description for port 0 on card 1):

```
*cli> setsnmp ifDescr.1000 ?
Enter an octet string
*cli> setsnmp ifDescr.1000
```

walksnmp

Use the **walksnmp** command to display the values of all MIB objects in the MIB tree starting with the specified object.

The **walksnmp** command displays the names and values of all variables that are "leaves" of the MIB tree below the specified MIB object. If you specify the root of a particular subtree, the command displays all of the "leaf" variables at the ends of branches in that subtree. If you specify an object that has no branches under it, the command displays the value of that object.

Use the **walksnmp** command to survey a range of variables, to locate a variable quickly when you know only which part of the MIB it is in, or to identify the name of a MIB variable so that you can specify it as an argument of another SNMP command.

Syntax

```
walksnmp mib-address
```

Argument

Use the *mib-address* argument to specify the starting point for the display. The address can be a path of dot-separated numbers, a variable name, or a combination of the two, where only the first element is a variable name. Thus, 1.3.6.1.2.1.1, mgmt.1.1, and system all refer to the same subtree. See the chapter entitled "LightStream 2020 MIB Reference" for a detailed description of MIB addresses. You can use the **walksnmp** command to identify MIB objects and their addresses.

Example

The following example shows how to use the **walksnmp** command to display MIB objects in the system subtree:

```
cli> walksnmp system
Name: sysDescr.0              Value: LightStream Data Switch
Name: sysObjectID.0           Value: lightStreamATM
Name: sysUpTime.0             Value: 26422638
Name: sysContact.0            Value: Lisa Bloch
Name: sysName.0               Value: Comet
Name: sysLocation.0           Value: Boston, 27/412
Name: sysServices.0           Value: 78
cli>
```