Installation & Operating Guide

FC1611/VFC1611 - 16 port fiber-cut switch

Echola Systems L.L.C
1161 Ringwood Ct, Ste 100
San Jose, CA-95131
Phone: 408 321 9663
Fax: 408 321 9663
http://www.echola.com

Installation

FC1611 is a Linux based 16 port fiber cut optical switch(16 1x1 switch in one unit). It provides a serial (RS232) and an Ethernet (10/100) port connectivity for management. Serial port is normally used in special situations such as to debug network connectivity if FC1611 is not reachable through Ethernet.

You might require a Laptop or a PC to configure FC611 with an IP address before connecting to your network so that you would be able to access FC1611 from remote. You could either use Serial or Ethernet port to configure IP. If you use serial port using supplied null modem cable then you would need to configure serial for **38400 baud rate with 8-N-1** to access FC1611. If you use Ethernet then you would need to configure the PC or Laptop's IP to match FC1611's default network. The default network configuration is as follows

IP Address: 10.1.1.100
Mask: 255.255.255.0
Gateway: 10.1.1.1
DNS: 10.1.1.1

If you have successfully configured Serial or Ethernet then you would see the os414 login prompt; login as "root" to setup network.

root's password: ***osctl***

There is one non-root default user available on FC1611 which can be used once the switch is setup (for doing switching of port).

username: *osctl*
password: *osctl*

Configuring Static IP

Use "osctl" command to configure a static IP address after login as "root" user. "osctl -?" shows detailed osctl command options with examples.

osctl -i 192.168.1.10 -m 255.255.255.0 -g 192.168.1.1

The above command configures IP address of FC1611 as 192.168.1.10 with mask 255.255.255.0 and gateway & DNS as 192.168.1. Once the IP is configured from a PC or a Laptop using Ethernet or serial port, you can then connect FC1611 to your network and access it using "telnet" or "ssh".

Configuring Dynamic IP

If you have a DHCP server running on your network and you may want to dynamically assign an available IP address to FC1611 by using following option.

```
# osctl -D
```

Make sure you know the assigned IP address to login using "telnet" or "ssh".

Configuring Hostname

You can also change the hostname of FC1611 switch using "osctl".

```
# osctl –h FC1611-SW-1
```

The above command changes hostname of FC1611 to FC1611-SW-1.

Configuring DNS/Nameserver

To configure a DNS or Nameserver, you can use "-n" option of osctl.

```
# osctl –n 192.168.1.11
```

Configuring Syslog Server

To send system generated events to an external syslog server, use "–S" option of osctl. You have to specify the address of the syslog server which will receive these event logs. (Note: this option only available in software version 2.0 and above).

```
# osctl –S 192.168.1.200
```

Version Info

The following command shows hardware and software versions and serial number of the switch.

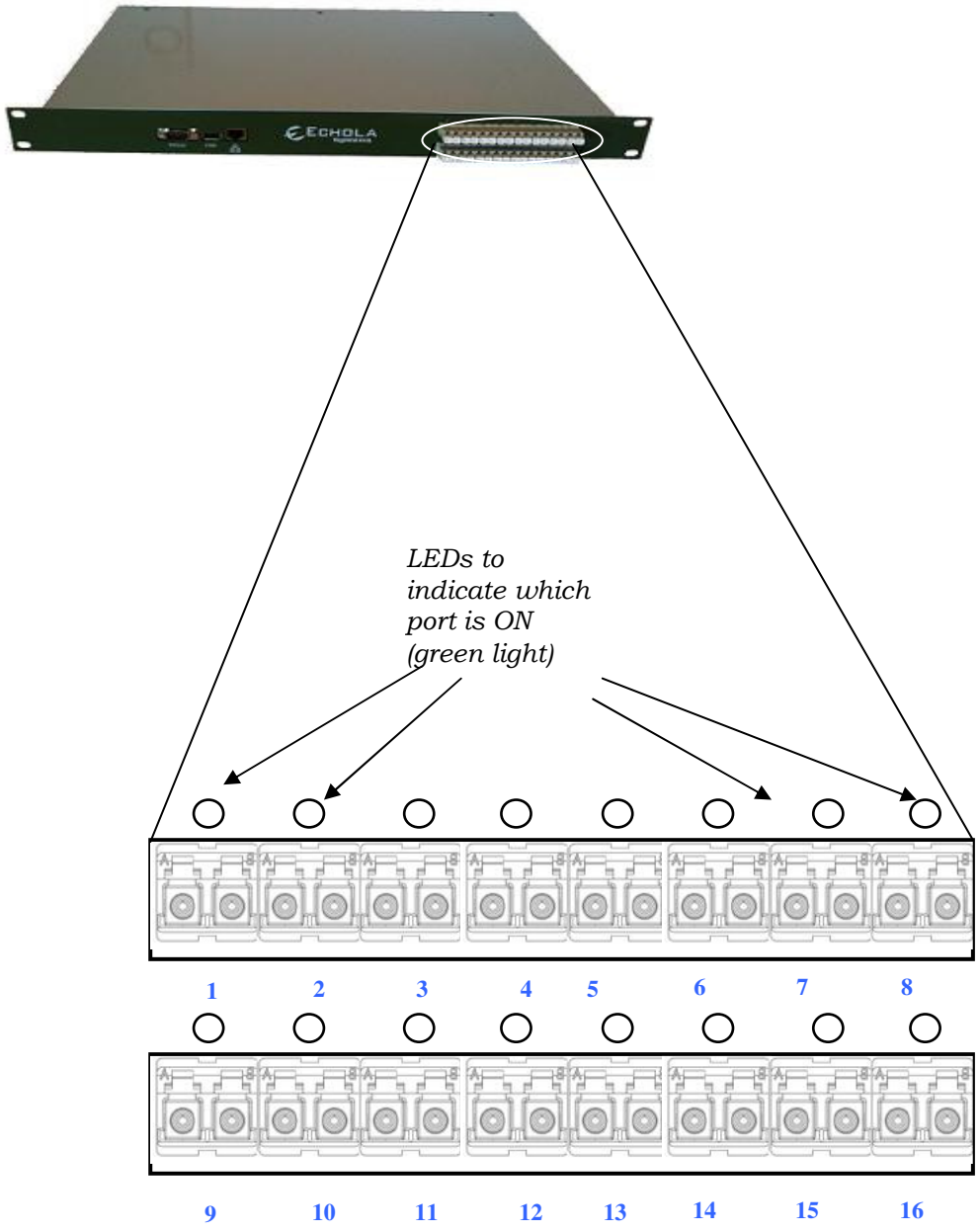
```
# osctl –v
```

Other Administrative commands

Most of other administrative functions can be done using standard Linux commands. For example, to change the password use "*passwd*" command from the Linux prompt and to add new user use "*useradd*" command. You have to be "root" user in order to add a new user.

Operation

FC1611 has sixteen individual 1x1 optical switches in single 1U enclosure. Each switch's input/output is connected to a duplex LC connector for external connectivity.



Each LC connector has two input/outputs marked as B and A (duplex). Suppose if you want to simulate fiber cut scenario between 2 optical devices.

```
                                Duplex Fiber
Device1-TX------------------------>----------------------Device2-RX
Device1-RX ------------------------<----------------------Device2-TX
```

Then you need to disconnect the fiber connecting these 2 devices which are under test and connect them through FC1611 ports, between A and B like the following,

Device1-TX -› {Port1B -› [FC1611 SWITCH_PORT1] -› Port1A} -› Device2-RX

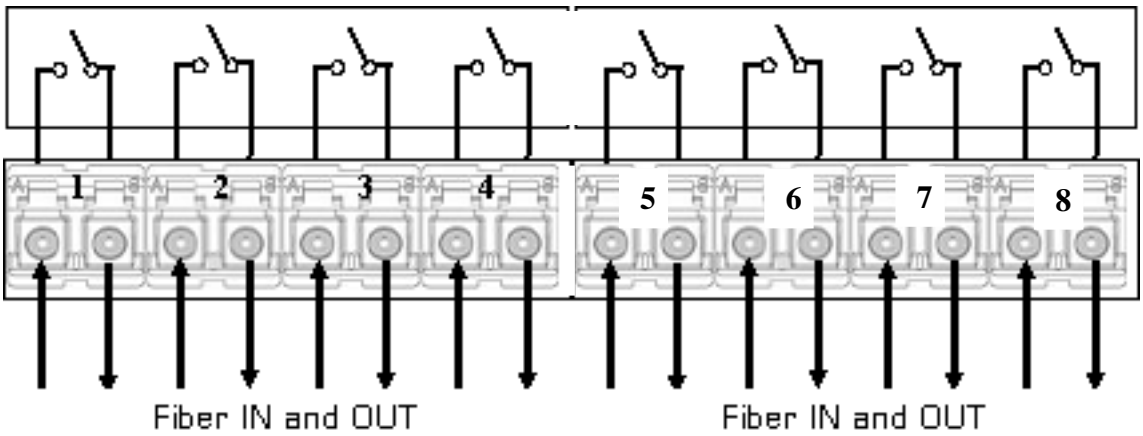
If you want to simulate break between both TX and RX fibers of the these devices under test (which may be the case in many scenarios) then you will have to use 2 ports of FC1611, like the following

Device1-TX -› {Port2B -› [FC1611 SWITCH_PORT1] -› Port2A} -› Device2-RX

Device1-RX ‹- {Port1B ‹- [FC1611 SWITCH_PORT2] ‹- Port1A} ‹-Device2-TX

UPSR, 2F-BLSR protection ring testing may require 2 FC1611 ports while 4F-BLSR might require 4 ports.

The following diagram shows how fiber cables should be connected to FC1611 (It shows only first 8 ports here).



Note that the 1x1 switch is between each LC port. You can have as many 8 pair of fiber cables connected through this device to simulate fiber-cut fault scenario.

Osctl - Command Line Interface

You can manage the FC1611 using a command line interface (CLI). To simply the operation, all functions are provided in a single command called "osctl". Osctl provides 3 major functions

1. *Network & Host configuration* (allowed only in "root")
2. *Switch Control & Status*
3. *Port group Management*

Osctl will show correct syntax usage if you make mistakes in typing command options. It will also provide some examples on how to use the command when you make mistakes. Also you may use "osctl -?" which shows all syntaxes with all examples.

Network & Host configuration

Following commands are used to configure network.

1. To configure static IP
   # osctl -i <ip> -m <mask> -g <gw>
2. To use DHCP
   # osctl -D
3. To configure hostname alone
   # osctl -h <hostname>
4. To configure DNS/NameServer
   # osctl -n <dns ip addr>
5. To configure Syslog server for event logging
   # osctl –S <syslog server ip addr>

For details refer to installation section as these are explained in that section.

Switch Control

In order to switch ON or OFF particular a port or ports or a group you can use following command.

$ **osctl -p {<port#/s> | <port_range> | <port_group>} <on|off> [-t <secs>]**

Note that the "{}" (braces) groups options and "|" is equivalent to "or". If the options are in square brackets "[]" then it is optional. Wherever you see port# or in_port# they all the same and represents one of 16 ports of FC1611. For example:

$ **osctl -p 4 on**

➔ switch port 4 to ON. LED on port 4 of FC1611 lids when this command is issued which confirms that the port is switched ON.

$ **osctl -p "1 2 3" on**

➔ switch ports 1, 2 and 3 to ON. Note the double quote, without that it won't work.

$ **osctl -p "1-4" off**

➔ switch ports 1, 2, 3 and 4 to OFF (range 1 to 4). This comes in handy when you want to switch bunch of ports in order.

$ **osctl -p gp1 off**

➔ switch all ports in group "gp1" to OFF. You will have to create group before using it with "-c" option as explained in "Group Management" section.

$ **osctl -p 2 off -t 120**

➔ switch port 2 to OFF after 120 **secs** (delayed switching)

$ **osctl -p 2 off -T 120**

➔ switch port 2 to OFF gradually with in 120 ***milliseconds***. This option is only available with VFC series switch. It is different from above delay where the switching happens almost immediately *after* 120 *secs of wait* but in this case switching

takes 120 milliseconds to come to completely OFF state to simulate real world fiber cuts. Default is 10 *milliseconds*.

Switch Status

Following command shows the status of a port or a group.

$ **osctl -s [<port_group> | <in_port#>]**

For example:

$ **osctl -s**

➔ shows all ports' & groups' status as follows. It shows first all ports and tells you which ports are part of a group.

```
        All Ports Status
====================================
    Port     | Port Status
====================================
    1       |    off
    2       |    off
    3       |    off
    4       |    off
    5       |    on
    6       |    on
    7       |    on
    8       |    on
  ……..
……….
……….
------------------------------------
        Group Status
================================
   Group Name  | Port Status
================================
  gp1        |    off
--------------------------------
==========================================
              Group's Port Details
==========================================
 GroupName: <gp1>
    Port Status: off
      Ports in the group: 1 2 3 4
------------------------------------------------------
```

$ **osctl -s gp1**

➔ shows group "gp1" status only. You will have to create group before using it with "-c" option as explained in "Group Management" section.

$ **osctl -s 2**

➔ shows port 2 status only

Port group Management

Port group management commands provide convenience of switching bunch of ports together identified by a name. Group name can be any alpha-numeric name. Group name can not be just a number or start with a number. For example group name can not be "10" or 10gp1. The following are the group commands. Whenever you see reference to port_group it is same as group name.

1. To create a new group
   $ **osctl -c <port_group> {<in_port#/s> | <inport_range>}**
2. To update existing group
   $ **osctl -u <port_group> {-a | -r <in_port#/s>}**
3. To delete a group
   $ **osctl -d <port_group>**
4. To delete all groups
   $ **osctl -R**
5. To list ports in a group
   $ **osctl -l [<port_group> | <in_port#>]**

For example:

$ **osctl -c gp1 "1 3 4"**
➔ creates group named "gp1" with ports 1,3 and 4. The double quotes around space separated port numbers are required, without that command will fail. Note that when you create a group, all the ports in the gp1 group are switched to OFF state by default.

$ **osctl -c gp2 "5-8"**
➔ creates group named "gp2" with ports 5,6,7 and 8 (5 to 8 range)

$ **osctl -u gp1 -a "5 6"**
➔ adds ports 5 & 6 to existing group gp1

$ **osctl -u gp1 -a 7**
➔ adds port 7 to existing group gp1

$ **osctl -u gp1 -r "2 4"**
➔ removes ports 2 & 4 from group gp1

$ **osctl -d gp1**
➔ deletes group gp1 and release all ports which were part of the group.

$ **osctl –l**
➔ lists ports in group, like

GroupName: <gp1>
Ports in the group: 1 2 3 4

$ **osctl –R**
➔ delete all groups in the database.

Setting Attenuation (VFC Series only)

Using VFC series switch as VOA

The "-A" option is used to set the attenuation of any particular output port or group to fraction of input power. It takes 1 to 256 number as the divisor. It sets the power of output port to **((256-*divisor*) * (*input power*/*divisor*))** *dB*. More the divisor, more the attenuation is (256 is fully off & 1 is on). Note: the attenuation setting is not saved, once you reset or reboot the switch this setting is lost; you will have to configure it again.

$ osctl -A { <port_group> | <in_port#> } { <divisor (1-256)> | [<divisor range(1-256)> -T <msec>] }

For example:

$ osctl -A 1 128
-> set attenuation of port#1 to 128 so that the output power is
-> ((256-128) * (input power/256)) dB => 1/2 the input power
-> More the divisor, more the attenuation (256 is fully off & 1 is on)

$ osctl -A gp1 64
-> set the attenuation group gp1ports to 64 so output power is
-> ((256-64) * (input power/256)) dB, which is 1/4 of the input

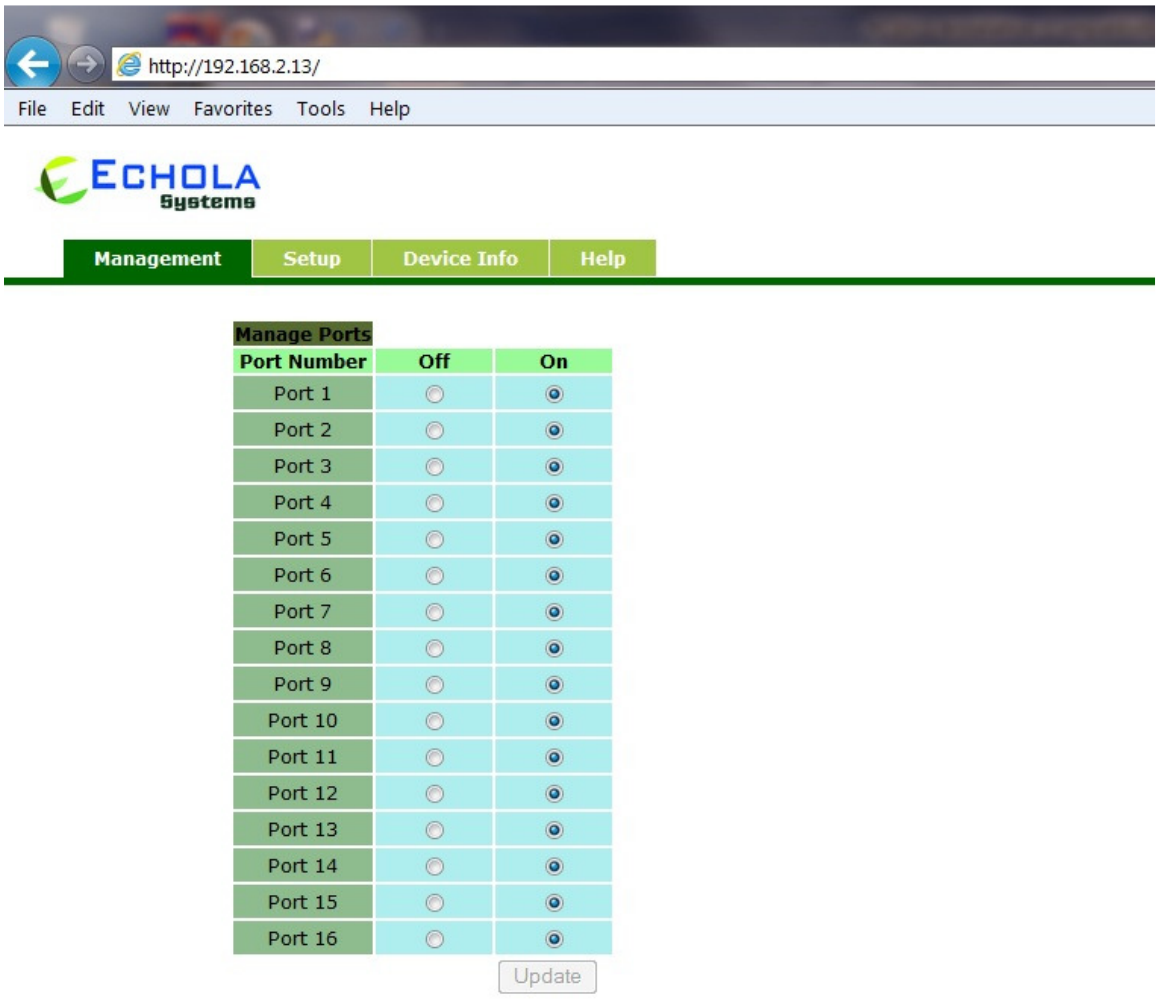
$ osctl -A 3-8 32
-> set the attenuation of ports 3,4,5,6,7 & 8 to 32
-> which is ((256-32) * (input power/256)) dB

$ osctl -A 1 100-160 -T 5000 (you need sw. ver 3.0 and above for this option)
-> set attenuation of port#1 to 100 first and ramp up to 160 in 5000 milliseconds period. This is useful for creating ***packet errors*** as opposed to fiber-cut.

$ osctl -A 1 160-100 -T 5000 (you need sw. ver 3.0 and above for this option)
-> set attenuation of port#1 to 160 first and ramp down to 100 in 5000 milliseconds period.

FC1611 Web Interface

In order to access FC1611 Web interface you will have to use its IP address in URL address bar of the web browser. The web interface is included in software version 2.2 and above. The first page you will see is the management page where you control all ports.



Automating Echola's Optical Switches

In order to automate Echola's Layer 1 switches you would need to write Tcl/Expect based scripts or Perl/Xml based scripts (both example scripts are given below). The tcl and expect scripting languages are easy to learn. We have given an example script written for FC1611/fc1611 at the end which you can modify to suit your need. There are tons of online sources for learning tcl & expect. The following provides quick high level overview of tcl and expect http://cplug.org/t/uploads/2009/02/tcl-expect.pdf. There is a good book from O'Reilly which provides great insight into expect language itself: "Exploring Expect: A Tcl-based Toolkit for Automating Interactive Programs (Nutshell Handbooks)".

Running scripts from Unix/Linux systems

If you want to run the script from a Unix/Linux based machines then there is possibility that you may be already having these tools on your system. Check if it's already been installed by typing "expect" from Unix/Linux prompt. If it is not then you will have to install it using package install tool for that particular flavor of Unix/Linux. For instance, on Fedore core Linux, you can use "yum install tcl expect" to install tcl and expect.

Running scripts from Windows

For windows based systems you can install windows free community version of ActiveTcl from Activestate http://www.activestate.com/activetcl/downloads. The expect is not available yet for 64bit version of Windows 7/Vista. So you will need to download 32bit version for ActiveTcl first and then make sure to install "expect" using command "teacup install Expect".
Also you need to enable "telnet" client on Windows before running any scripts. In order to enable telnet on Windows follow these steps

- ➔ Start
- ➔ Control Panel
- ➔ Programs And Features
- ➔ Turn Windows features on or off
- ➔ Check Telnet Client
- ➔ Hit OK

After that you can start Telnet via Command Prompt to check if it works.

The following sample script actually login into fc1611/FC1611 switch and issue a switch command then check whether the switch command was successful and return the result before terminating the telnet session. This script takes argument (port number and state of the port (on/off)) from commands line argument. Cut and Paste the following script on to any editor and save as "rosctl". Then you can run the script by issuing rosctl -p <port#> on|off. For instance, to switch port 2 to ON, you can call script as rosctl -p 2 on. Make sure you have proper path set for expect on first line "#!/usr/bin/expect" for Unix/Linux based systems. For windows you will have to uncomment ' exec' and 'package' commands as mentioned in the script. All comments inside '#' provide more info on what the script is doing.

Sample Tcl/Expect script

```
#!/usr/bin/expect
##########################################################
####################
# This script switches the given port and verifies if the port is switched from a
remote machine
#          Command Usage: rosctl -p <port#> on|off
##########################################################
####################
# For windows uncomment following
#    exec tclsh "$0" ${1+"$@"}
#    package require Expect

# Check number of arguments passed to this command if < 3 then spit out error &
exit

    if { $argc < 3 } {
        puts "Usage: rosctl -p <port#> on|off\n"
        exit 1
    }

# Set telnet host, username, password and other parameters, modify these to reflect
your setup

    set hostname "192.168.2.20"
    set username "osctl"
    set password "osctl"
    set prompt "osctl@.*\$"
    set port [lindex $argv 1]
    set status [lindex $argv 2]
    set commandcontrol "osctl -p $port $status"
    set commandstatus "osctl -s $port"

# Display info.

    puts "Connecting to $hostname."
```

```
# Connect to the telent server using the "spawn" command.

    spawn telnet $hostname
    #spawn C:\Putty\putty.exe -telnet $hostname


# Wait for a login prompt.

    expect -re "(Name|login|Login|Username).*:.*" {

      # Login prompt received. Send user name to FC1611/fc1611.

      send "$username\r"

    } eof {

      # No login prompt received. Display an error.

      puts "could not connect\n"
    }

# Wait for a password prompt from the Unix server.

    expect "Password:" {

      # Password prompt received. Send the password.

      send "$password\r"
    }

# Wait for the switch prompt.

    expect -re $prompt {

      # Issue osctl command to switch given port

      send "$commandcontrol\r"
    }

# Wait for the switch prompt again to check status.

    expect -re $prompt {

      # Issue osctl command to check status

      send "$commandstatus\r"
    }
```

```
# Discard echoed command - we need only the status

    expect "$commandstatus\r"

# Discard unwanted prompt as well

    expect -re "(.*)$prompt"

    #Debug
    #puts "\nGOT*****$expect_out(buffer)**************\n"
    #puts "\n GOTS ####$expect_out(1,string)#####\n"

# Save remaining to buffer 'data'

    set data $expect_out(1,string)

# Check return status and display result accordingly
    switch -re $data {
      "off" { puts "Port $port is OFF" }
          "on"  { puts "Port $port is ON" }
         default { puts "Port $port status is unknown" }
    }

# Terminate telnet

                                        send "exit\r"
```

Sample Perl/xml script

```
#########################################################################
#
# Command Syntax: osctlauto.pl http://<ipaddress> <port#> <on|off> #
#########################################################################
##
use LWP::UserAgent;
use XML::Simple;

$IPADDR = shift;
$PORT = shift;
$STAT = shift;

# create objects
$xml = new XML::Simple;
$ua = LWP::UserAgent->new;

# set status on/off

$REQUEST=$IPADDR . "/xmlapi/setstatus.php?port=" . $PORT . "&" . "status=" .
$STAT;
#print "REQUEST: $REQUEST\n";
$req = HTTP::Request->new(GET => $REQUEST);
$req->header('Cookie' => 'test=quest');
$res = $ua->request($req);

# check status

$REQUEST=$IPADDR . "/xmlapi/getstatus.php?port=" . $PORT;
print "REQUEST: $REQUEST\n";
$req = HTTP::Request->new(GET => $REQUEST);
$req->header('Cookie' => 'test=quest');
$res = $ua->request($req);
$data = $xml->XMLin($res->content);

#print status

print "STATUS = $data->{status}\n";
```

Troubleshooting

- ➔ If you use telnet and is very slow, it could be due to name server / DNS issue. Make sure you have name server configured correctly (-n option). Try "ping <name_server_ip>" to see if name server configured can be reached from OS414. If you don't have dns on your network then you can just remove file "/etc/resolv.conf" and see if it works normal.

- ➔ All the group information is stored in a hidden XML database file. If for some reason this file is corrupted, the system will recover from this error by trying to copy the backup database file. If this happens it will throw a warning message but it is not guaranteed that all the group information will be restored correctly. In that case you may need to recreate missing groups.

Hardware Specifications	
Electrical	
Input Power	100-240 AC
Total Power Consumption	< 7 Watts
Serial Port	1x RS232
Networking	1x 10/100 Ethernet
Optical Specification for FC Series	
Connectivity	16 duplex LC connectors
Data rate	Any data rate - Physical Layer Switch, no limitation
Wavelength (**nm**)	**Multimode version:** 850/1310 nm **Singlemode version:** 1280-1625 nm
Optical Technology	1x1 Electro-mechanical-optical switch
Insertion Loss (**dB**)	0.5 Typical, 0.8 Max. (without connectors)
Switching Time (**ms**)	≤ 10
Crosstalk (**dB**)	≤ -80
Repeatability (**dB**)	≤ 0.1
Optical Specification for VFC Series (SM)	
Connectivity	16 duplex LC connectors
Data rate	Any data rate - Physical Layer Switch, no limitation
Wavelength (**nm**)	1310±50nm & 1550±50nm
Optical Technology	MEMS VOA
Insertion Loss (**dB**)	< 0.8 dB. (without connectors)
Switching Time (**ms**)	≤ 5
Crosstalk (**dB**)	≥ 65
Attenuation Dynamic Range (**dB**)	> 30
Attenuation Resolution (**steps**)	256 steps
Optical Power Handling (**mW**)	≤ 250mW/channel
Environmental	
Operating Temperature (**°C**)	-5 ~ +75
Storage Temperature (**°C**)	-40 ~ +85
Relative Humidity Range (%)	0 ~ 85

Contact Info

If you have any technical questions and need help you can send email to support@echola.com or call 408-321-9663. You can also download latest documents and software from our website www.echola.com/optical.