



LightPulse[®] Utility (lputilnt)

for Windows Server 2003

User Manual

Last Updated December 4, 2007

Copyright© 2007 Emulex Corporation. All rights reserved worldwide. No part of this document may be reproduced by any means nor translated to any electronic medium without the written consent of Emulex Corporation.

Information furnished by Emulex Corporation is believed to be accurate and reliable. However, no responsibility is assumed by Emulex Corporation for its use; or for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Emulex Corporation.

Emulex, AutoPilot Installer, BlockGuard, cLAN, FabricStream, FibreSpy, Gigaset, HBAnyware, InSpeed, IntraLink, LightPulse, MultiPulse, SAN Insite, SBOD and Vixel are registered trademarks, and AutoPilot Manager, EZPilot, SLI and VMPilot are trademarks, of Emulex Corporation. All other brand or product names referenced herein are trademarks or registered trademarks of their respective companies or organizations.

Emulex provides this manual "as is" without any warranty of any kind, either expressed or implied, including but not limited to the implied warranties of merchantability or fitness for a particular purpose. Emulex Corporation may make improvements and changes to the product described in this manual at any time and without any notice. Emulex Corporation assumes no responsibility for its use, nor for any infringements of patents or other rights of third parties that may result. Periodic changes are made to information contained herein; although these changes will be incorporated into new editions of this manual, Emulex Corporation disclaims any undertaking to give notice of such changes.

Introduction.....	1
Installing Iputilnt.....	1
Running Iputilnt.....	1
Iputilnt Category Summaries.....	2
Discovering HBAs	3
Setting Driver Parameters	3
Activation Requirements.....	3
Setting Driver Parameters	3
Resetting HBA Values	5
Driver Parameter Reference Table	5
Setting Topology.....	10
Topology Reference Table.....	11
Mapping and Masking.....	12
Automapping SCSI Devices	12
Target and LUN Mapping and Masking.....	12
Overviews	12
Mapping and Masking Window Defaults.....	13
Setting Mapping and Masking	13
Globally Automapping All Targets	13
Globally Mapping All LUNs.....	14
Globally Unmasking or Masking All LUNs	14
Automapping LUNs for a Target	14
Setting Up Persistent Binding.....	15
Updating Firmware or Boot Code	17
Server Performance	18
I/O Coalescing.....	18
Performance Testing	19
QueueDepth	19
NumFcpContext	19
CoalesceMsCnt	19
CoalesceRspCnt.....	19
Performance Testing Examples.....	19
Test Scenario One	19
Test Scenario Two	19

Introduction

Use the LightPulse[®] utility (lputilnt) to do the following tasks on local HBAs:

- Download Peripheral Component Interconnect (PCI) configuration data files
- Assign an Arbitrated Loop Physical Address (AL_PA)
- Perform global and target mapping and masking
- Globally automap all logical unit numbers (LUNs)
- Globally unmask all LUNs
- Set up persistent binding
- Hot swap a device
- Set topology options
- Map device identifiers (IDs)
- Break SCSI reservations
- Set driver parameters
- Update firmware on the local HBA
- Enable boot code
- Update EFIBoot (64-bit only)

Installing lputilnt

Prerequisites

- The Storport Miniport driver is installed.
- The HBAnyware utility is installed.

To install lputilnt:

1. Download the lputilnt.zip file from the Emulex Web site to a directory on a local drive on the server.
2. Unzip the contents of the lputilnt.zip file.
3. Double-click Setup.exe to install.

Running lputilnt

To run lputilnt:

On the Windows desktop select **Start>All Programs>Emulex>LputilNT**.

Iputilnt Category Summaries

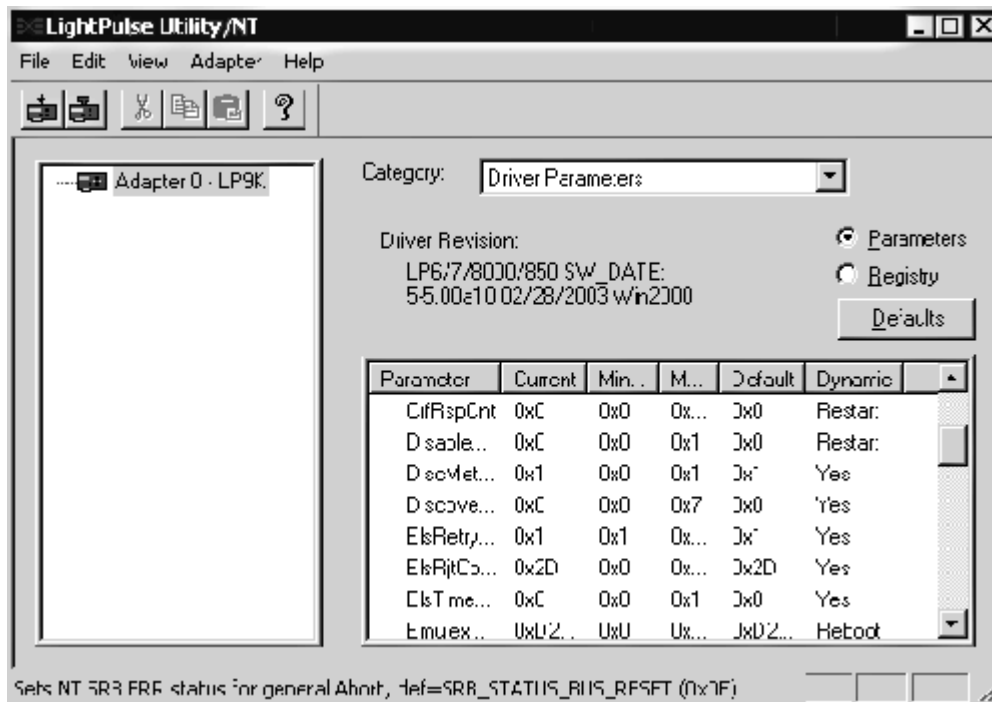


Figure 1: Driver Parameters view

Eight categories are available:

- Adapter revision levels - view information about the chipset and firmware revision levels of the selected HBA.
- Firmware maintenance - view details about the firmware in the flash read-only memory (ROM) of the selected HBA. Update HBA firmware and boot code, manage existing firmware and enable or disable the Boot bootup message.
- Loop map - view a list of the members of the selected HBA.
- PCI registers - view the values of the PCI configuration registers for the selected HBA.
- Configuration data - view information about the data in each of the configuration regions in the flash ROM of the selected HBA. Download PCI configuration files (CFL).
- Driver parameters - view and change device driver parameters.
- Persistent binding - view and manage persistent binding for the HBA, and LUN mapping and masking for devices in your SAN.
- Link statistics - view statistics about the arbitrated loop of the selected HBA.
- Status and counters - view status and counters for bytes, frames, sequences, exchanges, and so on.

Discovering HBAs

Local HBAs are discovered automatically when you start lputilnt.

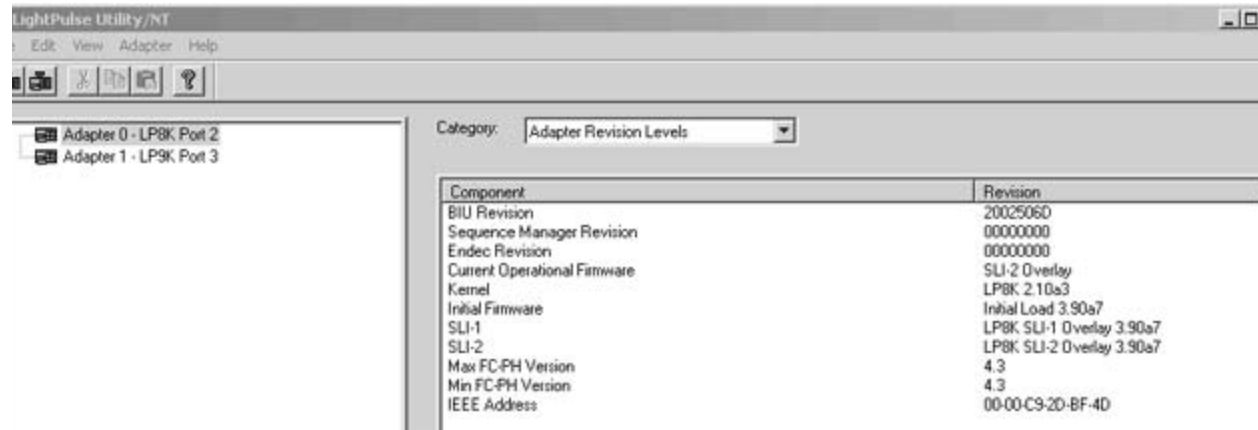


Figure 2: HBA Information (partial view)

Setting Driver Parameters

Activation Requirements

A parameter has one of the following activation requirements:

- Dynamic - the change takes effect while the system is running.
- Restart - requires an HBA reset from the utility before the change takes effect.
- Reboot - requires reboot of the entire machine before the change effect. In this case, you are prompted to do reboot when you exit the utility.

Setting Driver Parameters

Use lputilnt to change parameter values for the local HBA. You can also set all parameters back to the default value (out-of-box value) for the local HBA.

To change a driver parameter using lputilnt:

1. Start lputilnt.
2. Select an HBA.

3. Select Driver Parameters from the Category list.

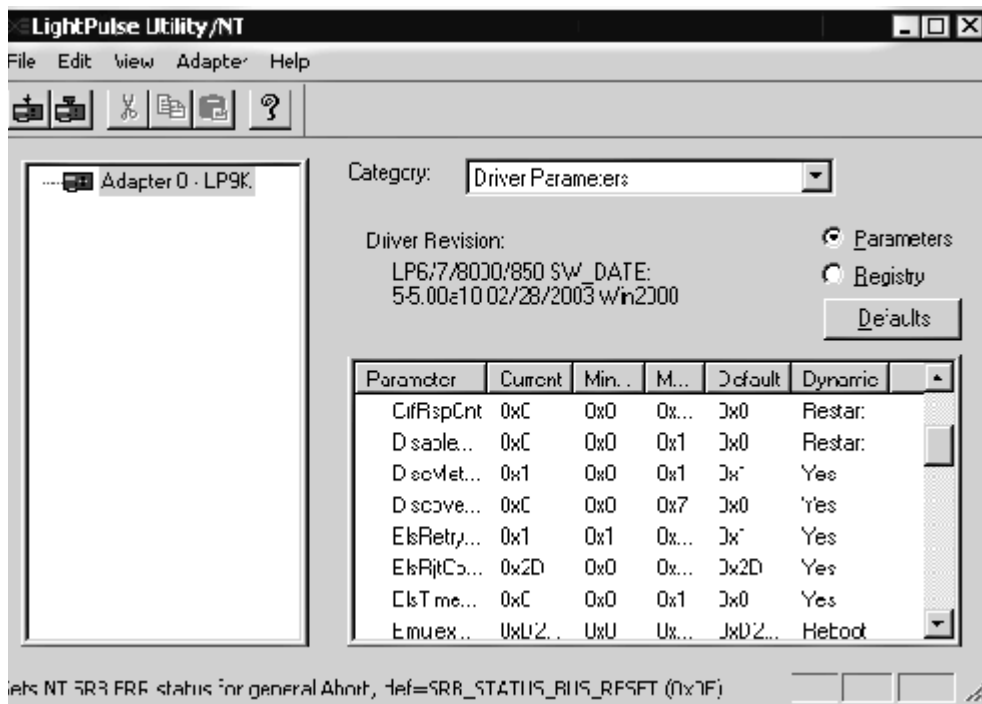


Figure 3: Driver Parameters view

4. Double-click the parameter to edit. The Modify Driver Parameter window is displayed.

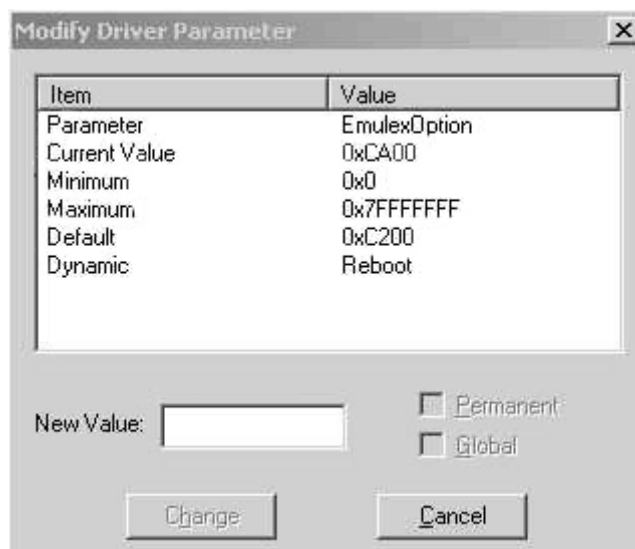


Figure 4: Modify Driver Parameter window

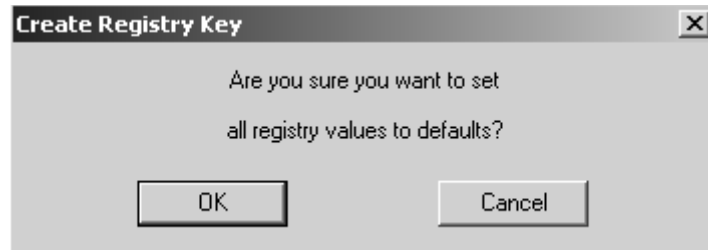
5. Enter a value in the New Value field in the same hexadecimal or decimal format as the current value. If the current value is in hexadecimal format, it is prefaced by "0x" (for example, 0x2d). You may enter a new hexadecimal value without the "0x". For example, if you enter ff10, this value is interpreted and displayed as "0xff10".
6. If desired and available, make the change permanent or global.
 - Select Permanent to write the new value to the system registry. If the Permanent box is not selected, the parameter reverts to its last permanent setting when the host is rebooted.

- Select Global to change the global registry entry. Otherwise, the change affects the selected HBA only.
7. Click **OK**.

Resetting HBA Values

To reset all the local HBA driver parameters back to their default (out-of-box) values:

1. Start Iputilnt.
2. Select an HBA.
3. Select Driver Parameters from the Category list.
4. Make sure that the Parameters radio button is selected and click **Defaults** (see Figure 2). The following window is displayed:



5. Click **OK** on the Create Registry Key window to set all parameters back to their defaults.

Driver Parameter Reference Table

Most parameters default to a setting that optimizes a typical operational scenario. If you are performance testing, see page 19 for recommended settings for the QueueDepth, NumFcpContext, CoalesceMsCnt and CoalesceRspCnt parameters.

Table 1: Driver Parameter Table

Parameter	Definition	Activation Requirement
AutoMap=n	<p>Controls the way targets are assigned SCSI IDs. Discovered targets are assigned persistent SCSI IDs (if configured by Iputilnt) according to the selected binding method. Persistent bindings do not take effect with the driver in stand-alone mode.</p> <p>If set to 0 = automap is disabled. Uses the LightPulse utility to persistently set the SCSI address of a discovered FCP capable FC node (target) into the registry. If set to 1 = automap by WWNN. If set to 2 = automap by WWPN. Allows WWPN type bindings to be saved in the registry using the LightPulse utility. If set to 3 = automap by DID).</p> <p>Value: 0 - 3 Default = 2</p>	Reboot

Table 1: Driver Parameter Table (Continued)

Parameter	Definition	Activation Requirement
Class= n	<p>Selects the class of service on FCP commands.</p> <p>If set to 2, class = 2. If set to 3, class = 3.</p> <p>Value: 2 - 3 Default = 3</p>	Dynamic
CoalesceMsCnt= n	<p>Specifies wait time in milliseconds to generate an interrupt response if CoalesceRspCnt has not been satisfied. Zero specifies an immediate interrupt response notification. A non-zero value enables response coalescing at the specified interval in milliseconds.</p> <p>Value: 0 - 63 (decimal) or 0x0 - 0x3F (hex) Default = 0 (0x0)</p>	Restart
CoalesceRspCnt= n	<p>Specifies the number of response entries that trigger an Interrupt response.</p> <p>Value: 1 - 255 (decimal) or 0x1 - 0xFF (hex) Default = 8 (0x8)</p>	Restart
DiscoveryDelay= n	<p>Controls whether the driver immediately starts port discovery or waits for 'n' seconds.</p> <p>If set to 0 = immediate discovery after link up. If set to 1 or 2 = the number of seconds to wait after link-up before starting port discovery.</p> <p>Value: 0 - 2 seconds (decimal) Default = 0</p>	Dynamic
EnableAck0= n	<p>Controls driver acknowledgement for class 2 traffic over an exchange. This applies to Fibre Channel Protocol (FCP) data exchanges on IREAD and IWRITE commands.</p> <p>If set to 0 = frame level acknowledgement. If set to 1 = forces sequence level acknowledgement.</p> <p>Value: 0 - 1 (decimal) Default = 0</p>	Restart
EnableAUTH = n	<p>Controls whether fabric authentication is enabled or disabled. This feature requires authentication to be supported by the fabric.</p> <p>If set to 0 = fabric authentication is disabled. If set to 1 = fabric authentication is enabled.</p> <p>Values: 0 - 1 Default = 0</p>	Reboot

Table 1: Driver Parameter Table (Continued)

Parameter	Definition	Activation Requirement
EnableFDMI= n	<p>Controls Fabric Device Management Interface (FDMI) functionality</p> <p>If set to 1 = management server login on fabric discovery is enabled and allows FDMI to operate on switches that have FDMI-capable firmware.</p> <p>If set to 2 = FDMI operates and uses the host name feature of FDMI.</p> <p>Value: 0 -2 (decimal) Default = 0</p>	Restart
EnableNPIV= n	<p>Controls whether N_Port_ID virtualization (NPIV) is enabled or disabled.</p> <p>If set to 0 = NPIV is disabled.</p> <p>If set to 1 = NPIV is enabled. Requires NPIV supported firmware and HBA.</p> <p>Value: 0 -1 Default = 0</p>	Restart
FrameSizeMSB= n	<p>Controls the upper byte of receive FrameSize if issued in PLOGI. This allows the FrameSize to be constrained on 256-byte increments from 256 (1) to 2048 (8).</p> <p>Value: 0 - 8 Default = 0</p>	Restart
HardALPA=0x n	<p>Allows the HBA to use a hard assigned loop address.</p> <p>Value: 0x00 - 0xEF (hex) Default = 0x00 (use soft addressing, or flash stored hard address value)</p> <p>Note: Only valid AL_PAs may be used.</p>	Restart
InitTimeout= n	<p>Determines the number of time-out seconds during driver initialization for the link to come up. If the link fails to come up by InitTmeout, driver initialization exits but is still successful. If the link comes up before InitTimeout, the driver sets double the amount for discovery to complete.</p> <p>Value: 5 -30 seconds or 0x5 - 0x1E (hex) Default = 15 seconds (0xF)</p>	Reboot

Table 1: Driver Parameter Table (Continued)

Parameter	Definition	Activation Requirement
LinkSpeed= n	<p>LinkSpeed has significance only if the HBA supports speeds other than one Gbit.</p> <p>If set to 0 = auto link speed detection. If set to 1 = 1 Gbit. If set to 2 = 2 Gbit. If set to 4 = 4 Gbit.</p> <p>Value: 0, 1, 2 and 4 Default = 0</p> <p>Note: Setting this option incorrectly may cause the HBA to fail to initialize.</p>	Restart
LinkTimeOut= n	<p>Applies to private loop only. With UseAdisc enabled, LinkTimeOut designates the number of seconds the driver waits between the time a link goes down and until all remaining logins will be invalidated (if the link is lost). Additionally, if UseAdisc is enabled and a value is assigned to LinkTimeOut, a timer is started on all mapped targets using the LinkTimeOut value. If the timer expires before discovery has completed, commands issued to timed out devices will return a SELECTION_TIMEOUT. The driver will be notified of a bus change event that will lead to the removal of all LUNs on the timed out devices.</p> <p>Value: 1 - 500 seconds or 0x0 - 0xFE (hex) Default = 30 (0x1E)</p> <p>Note: If UseAdisc is enabled (set to 1), LinkTimeOut is enabled. If UseAdisc is disabled (set to 0), LinkTimeOut is disabled.</p>	Dynamic
LogErrors= n	<p>Determines the minimum severity level required to enable entry of a logged error into the system event log. Error levels are severe, malfunction and command. A severe error requires user intervention to correct a firmware or HBA problem. A malfunction error indicates that the system has problems, but user intervention is not required. An object allocation failure is an example of a command error.</p> <p>If set to 0, all errors regardless of severity are logged. If set to 1, command level errors are logged. If set to 2, malfunction errors are logged. If set to 3, severe errors are logged.</p> <p>Value: 0 - 3 Default = 3</p>	Dynamic
NetworkOption= n	<p>Controls whether IP over FC is disabled or enabled. A value of 1 will enable IP over FC and will allow first time installation or startup of the FC LAN driver.</p> <p>Value: 0 - 1 Default = 0</p>	

Table 1: Driver Parameter Table (Continued)

Parameter	Definition	Activation Requirement
NodeTimeout=n	<p>The node timer starts when a node (i.e. discovered target or initiator) becomes unavailable. If the node fails to become available before the NodeTimeout interval expires, the OS is notified so that any associated devices (if the node is a target) can be removed. If the node becomes available before NodeTimeout expires the timer is canceled and no notification is made.</p> <p>Value: 1 - 255 seconds or 0x0 - 0xFF (hex) Default = 30 (0x1E)</p>	Dynamic
PciMaxRead=n	<p>Enables override of default PCI read transfer length. The driver will auto-detect the presence of an AMD PCI bridge and adjust for this bridge. This parameter allows for override of the automatic value.</p> <p>Values: 512, 1024, 2048, 4096 Default = 2048</p>	Restart
QueueDepth=n	<p>QueueDepth requests per LUN/target (see QueueTarget parameter). If you expect the number of outstanding I/Os per device to exceed 32, then you must increase to a value greater than the number of expected I/Os per device (up to a value of 254). If the QueueDepth value is set too low, a performance degradation can occur due to driver throttling of its device queue.</p> <p>Value: 1 - 254 or 0x1 - 0xFE (hex) Default = 32 (0x20)</p>	Dynamic
QueueTarget=n	<p>Controls I/O depth limiting on a per target or per LUN basis.</p> <p>If set to 0 = depth limitation is applied to individual LUNs. If set to 1 = depth limitation is applied across the entire target.</p> <p>Value: 0 -1 or 0x0 - 0x1 (hex) Default = 0 (0x0)</p>	Dynamic
RmaDepth=n	<p>Sets the remote management buffer queue depth. The greater the depth, the more concurrent management controls can be handled by the local node.</p> <p>Value: 8 - 64, or 0x8 - 0x40 (hex) Default = 16 (0x10)</p> <p>Note: The RmaDepth driver parameter pertains to the functionality of the HBAnyware utility.</p>	Reboot

Table 1: Driver Parameter Table (Continued)

Parameter	Definition	Activation Requirement
ScanDown= n	<p>Applies to private loop only in D_ID mode.</p> <p>If set to 0 = lowest AL_PA = lowest physical disk (ascending AL_PA order).</p> <p>If set to 1 = highest AL_PA = lowest physical disk (ascending SEL_ID order).</p> <p>Value: 0 - 1 Default = 0</p>	Reboot
SLImode= n	<p>Auto firmware selection mode uses SLI3 mode whenever the firmware and the driver agree that this is possible.</p> <p>If set to 2 = implies running the HBA firmware in SLI-2 mode.</p> <p>If set to 0 = autoselect firmware, use the newest firmware installed.</p> <p>Value: 0 - 2 Default = 0</p>	Reboot
Topology= n	<p>Topology values may be 0 to 3.</p> <p>If set to 0 (0x0) = FC-AL (loop).</p> <p>If set to 1 (0x1) = PT-PT fabric.</p> <p>If set to 2 (0x2) = *FC-AL first, then attempt PT-PT.</p> <p>If set to 3 (0x3) = *PT-PT fabric first, then attempt FC-AL.</p> <p>* Topology fail-over requires v3.20 firmware or higher. If firmware does not support topology fail-over, options 0,2 and 1,3 are analogous.</p> <p>Value: 0 - 3 Default = 2 (0x2)</p>	Restart
TraceBufSiz= n	<p>Sets the size in bytes for the internal driver trace buffer. The internal driver trace buffer acts as an internal log of the driver's activity.</p> <p>Value: 250,000 - 2,000,000 or 0x3D090 - 0x1E8480 (hex). Default = 250,000 (0x3D090)</p>	Reboot

Setting Topology

To change topology:

1. Select an HBA.
2. Select Driver Parameters from the Category list.
3. Double-click on **Topology** and the Modify Driver Parameter window is displayed.
4. Enter new topology value in the New Value field and click **Change**.
5. Reset the HBA to make this change effective.

Topology Reference Table

The presence of a fabric is detected automatically.

Table 2: Topology Reference

Topology	Description	Iputilnt Value
Private Loop Operation	<p>Only Fibre Channel arbitrated loop (FC-AL) topology is used. After successful loop initialization, the driver attempts login with FL_PORT (Switched Fabric Loop Port).</p> <ul style="list-style-type: none"> • If FL_PORT login is successful, public loop operation is employed. • If FL_PORT login is unsuccessful, private loop mode is entered. If a fabric is not discovered and the topology is arbitrated loop, the driver operates in private loop mode using the following rules: <ul style="list-style-type: none"> • If an FC-AL device map is present, each node described in the map is logged and verified as a target. • If an FC-AL device map is not present, logins are attempted with all 126 possible FC-AL addresses. LPGO/PRLO are also handled by the driver. Reception of either causes a new discovery or login to take place. 	0
Switched Fabric Operation	<p>Only if switched F_PORT(point-to-point [pt.-to-pt.]) login is successful, fabric mode is used.</p> <ul style="list-style-type: none"> • If F_PORT login is unsuccessful, N_PORT-to-N_PORT direct connection topology will be used. • If a switch is discovered, the driver performs the following tasks: <ul style="list-style-type: none"> • FL_PORT login (Topology = 0;) or F_PORT login (Topology =1;). • Simple Name Server login. • State Change Registration. • Symbolic Name Registration. • FCP Type Registration if RegFcpType is set to 1. • The driver logs out and re-logs in. The name server indicates that registration is complete. • Simple Name Server Query for devices (the registry parameter SnsAll determines if all N_Ports are requested (SnsALL=1;) or only SCSI FCP N_Ports (SnsAll=0; default) • Discovery/device creation occurs for targets described by the Name Server. • The driver handles RSCN and LOGO/PRLO. Reception of either causes new discovery/logins to take place. 	1
*FC-AL attempt first, then attempt pt.-to-pt.	<ul style="list-style-type: none"> • Topology fail-over requires v3.20 firmware or higher. If firmware does not support topology fail-over, options 0 and 2 are analogous. Options 1 and 3 are analogous. 	2
*pt.-to-pt. fabric attempt first, then attempt FC-AL.	<ul style="list-style-type: none"> • Topology fail-over requires v3.20 firmware or higher. If firmware does not support topology, fail-over options 0 and 2 are analogous. Options 1 and 3 are analogous. 	3

1: This driver is "Soft-Zone-Safe".

2: In a fabric environment, the order that disk devices are created is based upon the name server response data (which is not guaranteed in any special order). Between successive boots, the same device may be identified with a different physical device number. However, any devices which have been assigned a device letter through disk administrator continue to use that letter regardless of the physical device number.

Mapping and Masking

Automapping SCSI Devices

The driver defaults to automatically mapping SCSI devices. This procedure applies if the default has been changed.

To automap SCIS devices:

1. Run `lputilnt`.
2. Select an HBA.
3. Select Driver Parameters from the Category list.
4. Double-click on Automap and the Modify Driver Parameter window is displayed:
5. Enter new automap value in the New Value field and click **Change**.
6. Reboot the system for this change to take effect.

Target and LUN Mapping and Masking

Overviews

Globally Automapping All Targets

Global Automap All Targets defaults to enabled to allow the Emulex driver to detect all FC devices attached to the Emulex HBAs. Global automapping assigns a WWPN, target ID, SCSI bus and SCSI ID to the device. The SCSI bus and SCSI ID may change when the system is rebooted. With persistent binding applied to a target, the SCSI bus and SCSI ID remain the same, whether the system is rebooted or Global Automap All Targets is enabled. With Global Automap All Targets disabled, the Emulex driver detects FC devices attached to the HBA, and does not pass them to the operating system unless they have already been persistently bound.

Globally Automapping All LUNs

Global Automap All LUNs defaults to enabled and assigns an operating system LUN ID to a FC LUN ID for all LUNs behind the targets in your SAN. LUN mapping can also be enabled and disabled at the target level. Global automapping of LUNs is different from persistent binding. Global LUN automapping does not concern itself with the SCSI ID or SCSI Bus because the global LUN mapping will stay the same for the target when the system reboots.

Globally Unmasking All LUNs

Globally Unmask All LUNs defaults to enabled, to allow the operating system to see all LUNs behind all targets. If Globally Unmask All LUNs is set to disabled and you want the operating system to see the LUNs behind a specific target, you need to set unmasking at the target level.

Target LUN Automapping

Target LUN automapping defaults to disabled. If enabled, target LUN automapping assigns operating system LUN IDs to a fixed FC target's physical LUNs. Global LUN automapping must be disabled to do target LUN automapping. Target LUN automapping differs from persistent binding: persistent binding assigns a WWPN of a FC target device to an operating system target ID, SCSI bus and SCSI ID. The SCSI bus and SCSI ID may change when the system is rebooted. With persistent binding applied to one of these target devices, the SCSI bus and SCSI ID remain the same when the system is rebooted or global target automapping is disabled. LUN paths are displayed in Disk Manager (when you perform a re-scan).

Target LUN Masking

Target LUN masking defaults to disabled. You can mask and unmask LUNs at the target level. If you have unmasked all LUNs for a specific target (using global or target functions), you can mask and unmask an individual LUN as well. The HBA can detect all LUNs for a specific target and will present only the unmasked ones.

Mapping and Masking Window Defaults

Table 3 describes LUN mapping and masking global defaults.

Table 3: Mapping and Masking Window Defaults

Field (Function)	Default	Description	Window
Globally Automap All Targets	Enabled	Emulex driver detects all FC devices attached to the Emulex HBAs.	Global Automap
Globally Automap All LUNs	Enabled	Assigns an operating system LUN ID to a FC LUN ID for all LUNs behind all targets in the system area network.	Global Automap
Globally Unmask All LUNs	Enabled	Allows the operating system to see all LUNs behind all targets.	Global Automap
Automap All LUNs (Target Level)	Disabled	With Globally Automap All LUNs disabled, this parameter assigns an operating system LUN ID to a FC LUN ID for all LUNs behind the selected target.	LUN Mapping
LUN Unmasking (Target Level)	Disabled	Allows the operating system to see all LUNs behind the selected target. With this parameter disabled, each individual LUN can be masked or unmasked.	LUN Mapping

Setting Mapping and Masking

The driver defaults to enabling global mapping and masking tasks. The procedures in this section apply if the default has been changed.

Prerequisites

- The Storport Miniport driver is installed.
- The HBAnyware utility is installed.
- The lputilnt utility is installed.
- A target device with LUNs is properly configured.
- For automapping LUNS for a target, the Global Automap All LUNs setting on the Global Automap window is disabled. If necessary, disable this function and reboot the system before automapping LUNS for a target.

Globally Automapping All Targets

Global Automap All Targets defaults to enabled to allow the Emulex driver to detect all FC devices attached to the Emulex HBAs.

To globally automap all targets:

1. Run lputilnt.
2. Select an HBA.
3. Select Persistent Bindings from the Category list.

4. Click **Automap**. The Global Automap window is displayed.
5. Change the Automap All Targets setting to Enabled.
6. Click **OK**. The window closes.
7. Reboot the system for this change to take effect.

Note: With persistent binding applied to one of these targets, the SCSI bus and SCSI ID remain the same when the system is rebooted.

Globally Mapping All LUNs

Global Automap All LUNs defaults to enabled and assigns an operating system LUN ID to a FC LUN ID for all LUNs behind the targets in the SAN. LUN mapping can be enabled/disabled at the target level.

To globally map all LUNs:

1. Run `lputilnt`.
2. Select an HBA.
3. Select Persistent Bindings from the Category list.
4. Click **Automap**. The Global Automap window is displayed.
5. Change the Globally Automap All LUNs setting to Enabled.
6. Click **OK**. The window closes.
7. Reboot the system for this change to take effect.

Globally Unmasking or Masking All LUNs

Globally Unmask All LUNs defaults to allow the operating system to see all LUNs behind targets.

To globally unmask or mask all LUNs:

1. Run `lputilnt`.
2. Select an HBA.
3. Select Persistent Bindings from the Category list.
4. Click **Automap**. The Global Automap window is displayed.
5. Change the Unmask All LUNs setting to Enabled.
6. Click **OK**. The window closes.

This change takes effect without a reboot.

Automapping LUNs for a Target

Target LUN Automapping defaults to disabled. If enabled, target LUN automapping assigns an operating system LUN ID to a fixed FC target's physical LUN.

To automap LUNs for a target:

1. Run `lputilnt`.
2. Select an HBA.
3. Select Persistent Bindings from the Category list. All targets are displayed.
4. Click on a target. The **Lunmap** button becomes active.
5. Click **Lunmap**. The LUN Mapping window is displayed.
6. Change the LUN Automap function to Enabled. Target automapping assignment occurs and these assignments are displayed on the LUN Mapping window.

7. Click **OK**.
8. Reboot the system for this change to take effect.

LUN Masking and Unmasking for a Target

Target LUN Automapping defaults to disabled. If enabled, target LUN automapping assigns operating system LUN IDs to a fixed FC target's physical LUNs. Global LUN automapping must be disabled for target LUN automapping.

To unmask or mask a LUN:

1. Run `lputilnt`.
2. Select an HBA.
3. Select **Persistent Bindings** from the **Category** list. All targets are displayed.
4. Click on a target. The **Lunmap** button becomes active.
5. Click **Lunmap**. The LUN Mapping window is displayed. All LUNs are displayed for the target.
6. Do the following:
 - To unmask or mask all LUNs for the target, set the LUN Unmasking function to Disabled.
 - To mask or unmask a LUN, select the row and click **Edit**. In the Edit Map Entry area, click on the Mask (Unmask) field to change the status.

Note: If LUNs are not displayed, LUN mapping has been disabled at the global level and not enabled at the target level, or the LUNs have been masked at the global level.

7. Click **OK**.

These changes do not require a reboot.

Setting Up Persistent Binding

Global automapping assigns a binding type, target ID, SCSI bus and SCSI ID to the device. The binding type, SCSI bus and SCSI ID may change when the system is rebooted. With persistent binding applied to one of these targets, the WWPN, SCSI bus and SCSI ID remain the same, whether the system is rebooted or with Global Automap All Targets subsequently disabled (enabled by default). The binding information is permanent because it is stored in the Windows registry. The driver refers to the binding information at bootup.

Persistent binding permanently maps a device to the following:

- Binding type
- SCSI bus
- SCSI ID

`lputilnt` allows you to set up persistent binding on local HBAs only.

To perform binding tasks:

1. Run `lputilnt`.
2. Select an HBA.

3. Select **Persistent Bindings** from the **Category** list. All targets are displayed.

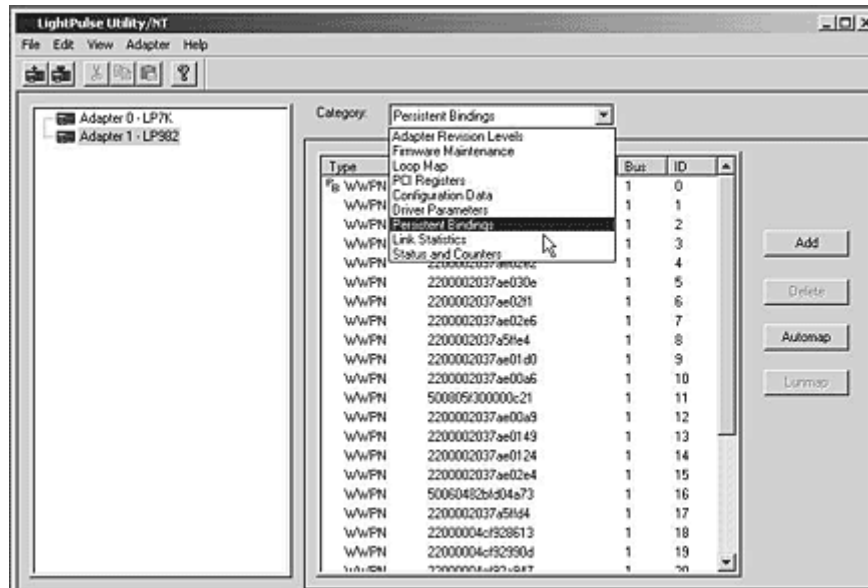


Figure 5: Iputilnt Utility, Persistent Bindings Category

4. Click on a target and click **Add**. The Add Binding window is displayed.



Figure 6: Iputilnt Utility, Add Binding window

5. In the Add Binding window, click the target you want to bind in the Unbound Targets list.
6. The Automap driver parameter controls assigned SCSI IDs. Discovered targets are assigned persistent SCSI IDs, according to the selected binding method. Persistent bindings do not take effect with the driver in stand-alone mode.
 - If set to 0 = automap is disabled. Uses the driver utility to persistently set the SCSI address of a discovered FCP capable FC node (target) into the registry.
 - If set to 1 = automap by WWNN.
 - If set to 2 = automap by WWPNN.
 - If set to 3 = automap by DID).

Default = 2

See “Setting Driver Parameters” on page 3 for information on how to set these driver parameters.

7. If necessary, change the SCSI Bus and SCSI ID values.

8. Click **OK** to bind the target. The letters "PB" will be displayed next to the target row.

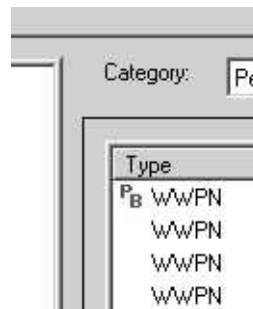


Figure 7: Persistent Binding Designation for Targets

9. Reboot the system for these changes to take effect.
10. Run `lputilnt`. Your new device and SCSI ID mapping information is displayed in the SCSI Target List area.

Updating Firmware or Boot Code

`lputilnt` allows you to update firmware or boot code on local HBAs only.

Prerequisites

- The Storport Miniport driver is installed.
- The HBAAnyware utility is installed.
- The `lputilnt` utility is installed.
- The firmware or boot code file has been downloaded from the Emulex Web site and extracted to a directory on your local drive.
- The system is in a state safe in which this type of maintenance can be performed:
 - I/O activity on the bus has been quieted.
 - Cluster software, or any other software that relies on the HBA to be available, has been stopped or paused.

Caution: Firmware or boot code versions differ between HBA models. Make sure you have downloaded the appropriate firmware or boot code for your HBA.

Note: No other functions can be performed while firmware or boot code loading is in progress.

To update firmware or boot code:

1. Run `lputilnt`.
2. Select the desired HBA.
3. Select Firmware Maintenance from the Category list.

Note: If the letter *W* appears next to a firmware entry, the image is represented in the wakeup parameters. This means that the HBA will use that specific image if it needs a firmware image.

4. Click **Download**.
5. Locate the new firmware or boot code file.
6. Click **Open**.
7. File downloading begins.

If you are updating firmware or boot code on a dual-channel HBA, repeat steps 2 - 6 to update the firmware or boot code on a second port.

8. If you are updating boot code, you must also enable the HBA to boot from SAN using the BIOS utility; see the documentation that accompanies the boot code for more information.
9. From the **Control Panel**, select **Security Center**. The Windows Security Center window

Server Performance

I/O Coalescing

I/O Coalescing is enabled and controlled by two driver parameters: CoalesceMsCnt and CoalesceRspCnt. The effect of I/O Coalescing will depend on the CPU resources available on the server. With I/O Coalescing turned on, interrupts are batched, reducing the number of interrupts and maximizing the number of commands processed with each interrupt. For heavily loaded systems, this will provide better throughput.

With I/O Coalescing turned off (the default), each I/O processes immediately, one CPU interrupt per I/O. For systems not heavily loaded, the default will provide better throughput. The following table shows recommendations based upon the number of I/Os per HBA.

Table 4: Recommended Settings for I/O Coalescing

I/Os per Second	Suggested CoalesceMsCnt	Suggested CoalesceRspCnt
I/Os < 10000	0	8
10000 < I/Os < 18000	1	8
18000 < I/Os < 26000	1	16
I/Os > 26000	1	24

CoalesceMsCnt

The CoalesceMsCnt parameter controls the maximum elapsed time in milliseconds that the HBA waits before it generates a CPU interrupt. The value range is 0 - 63 (decimal) or 0x0 - 0x3F (hex). The default is 0 and disables I/O Coalescing.

CoalesceRspCnt

The CoalesceRspCnt parameter controls the maximum number of responses to batch before an interrupt generates. If CoalesceRspCnt expires, an interrupt generates for all responses collected up to that point. With CoalesceRspCnt set to less than 2, response coalescing is disabled and an interrupt triggers for each response. The value range for CoalesceRspCnt is 1 - 255 (decimal) or 0x1 - 0xFF (hex). The default value is 8.

Note: A system restart is required to make changes to CoalesceMsCnt and/or CoalesceRspCnt.

Performance Testing

There are four driver parameters that need to be considered (and perhaps changed from the default) for better performance testing: QueueDepth, NumFcpContext, CoalesceMsCnt and CoalesceRspCnt.

Note: Parameter values recommended in this topic are for performance testing only and not for general operation.

QueueDepth

If the number of outstanding I/Os per device is expected to exceed 32, increase this parameter to a value greater than the number of expected I/Os per device, up to a maximum of 254. The QueueDepth parameter defaults to 32. If 32 is set and not a high enough value, performance degradation may occur due to Storport throttling its device queue.

NumFcpContext

If the number of outstanding I/Os per HBA is expected to exceed 512, increase this parameter to a value greater than the number of expected I/Os per HBA. Increase this value in stages: from 128 to 256 to 512 to 1024 to a maximum of 2048i. NumFcpContext limits the number of outstanding I/Os per HBA, regardless of how QueueDepth is set. The NumFcpContext defaults to 512. If NumFcpContext is too small relative to the total number of outstanding I/Os on all devices combined, performance degradation may occur due to I/O stream throttling.

CoalesceMsCnt

CoalesceMsCnt defaults to zero. If you are using a performance evaluation tool such as IOMETER and if you expect the I/O activity will be greater than 8000 I/Os per second, set CoalesceMsCnt to 1 and re initialized with an HBA reset or system reboot.

CoalesceRspCnt

CoalesceRspCnt defaults to 8. For all other values up to the maximum of 63, the HBA will not interrupt the host with a completion until either CoalesceMsCnt milliseconds has elapsed or CoalesceRspCnt responses are pending. The value of these two driver parameters reduces the number of interrupts per second which improves overall CPU utilization. However, there is a point where the number of I/Os per second is small relative to CoalesceMsCnt and this will slow down the completion process, causing performance degradation.

Performance Testing Examples

Test Scenario One

You execute IOMETER with an I/O depth of 1 I/O per device in a small-scale configuration (16 devices). In this case, the test does not exceed the HBA's performance limits and the number of I/Os per second are in the low thousands.

Recommendation: set CoalesceMsCnt to 0 (or leave the default value).

Test Scenario Two

You execute IOMETER with an I/O depth of 48 per device in a small-scale configuration (16 devices).

Recommendation: set QueueDepth to be greater than 48 (e.g. 64) and NumFcpContext to be greater than 512 (e.g. 1024).