SPC BENCHMARK 1™
FULL DISCLOSURE REPORT

FUJITSU LIMITED
FUJITSU STORAGE SYSTEMS ETERNUS DX200 S3

SPC-1 V1.14

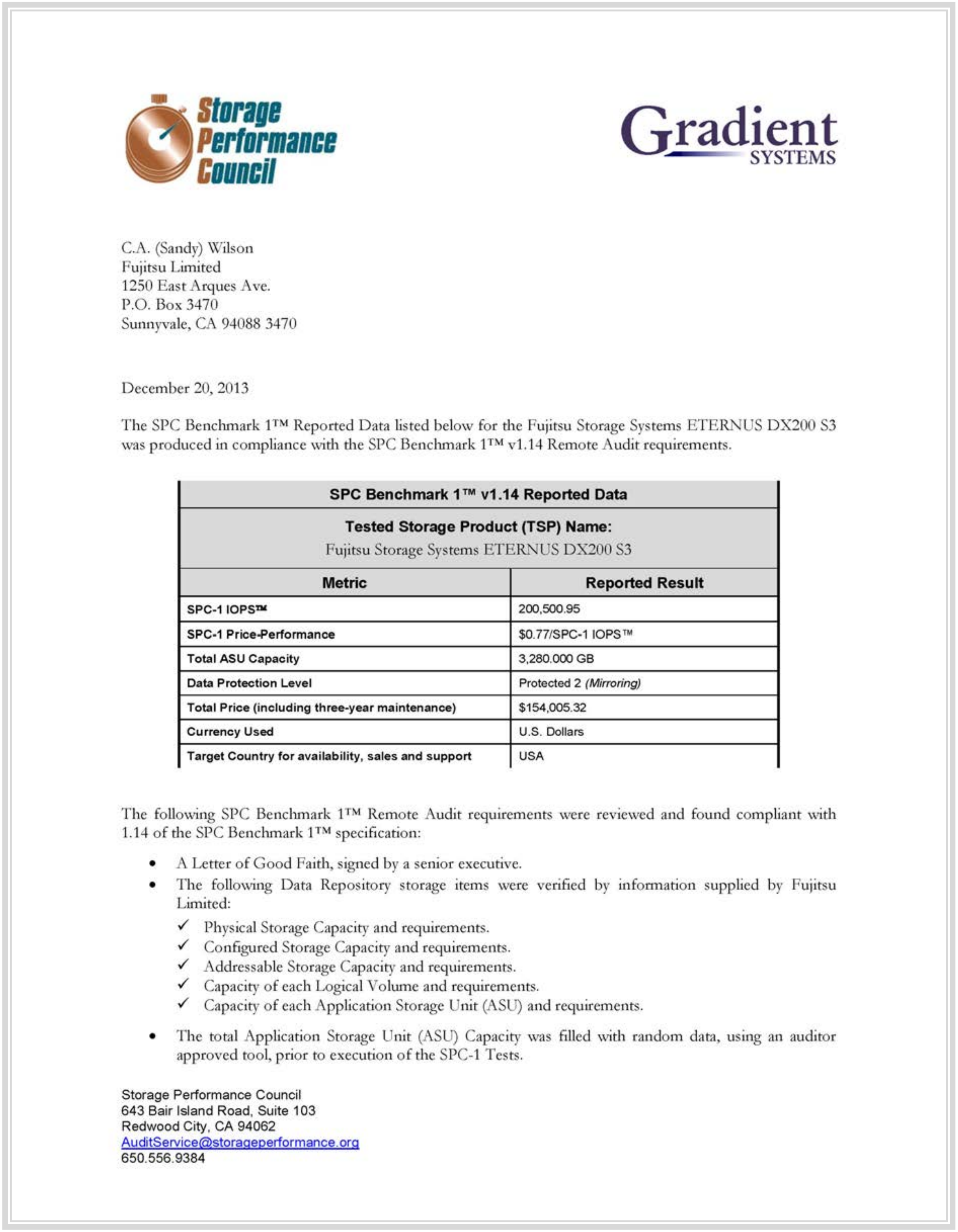
Submitted for Review: December 30, 2013
Submission Identifier: A00139

First Edition – December 2013

Trademarks

Table of Contents

AUDIT CERTIFICATION



C.A. (Sandy) Wilson
Fujitsu Limited
1250 East Arques Ave.
P.O. Box 3470
Sunnyvale, CA 94088 3470

December 20, 2013

The SPC Benchmark 1™ Reported Data listed below for the Fujitsu Storage Systems ETERNUS DX200 S3 was produced in compliance with the SPC Benchmark 1™ v1.14 Remote Audit requirements.

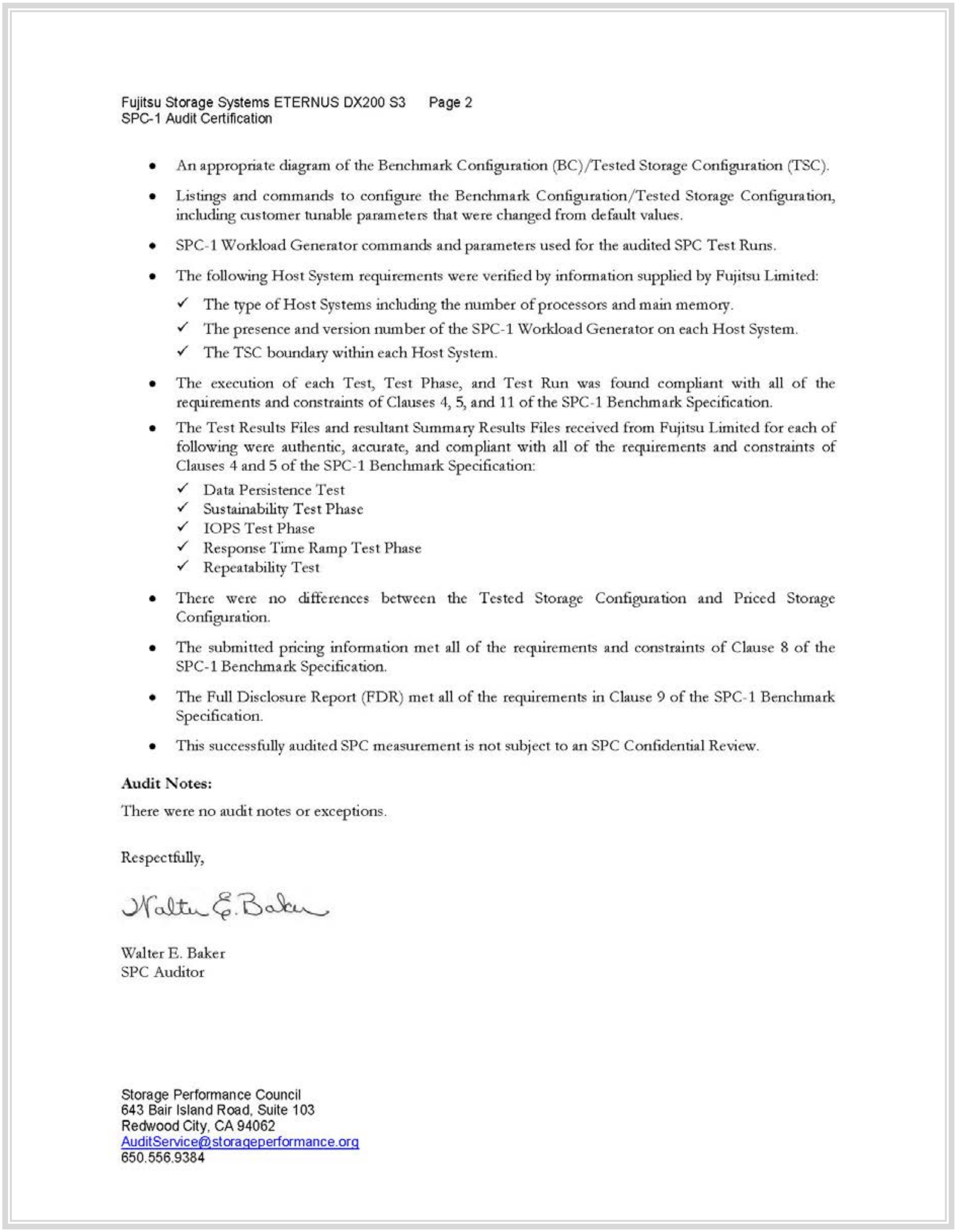
SPC Benchmark 1™ v1.14 Reported Data	
Tested Storage Product (TSP) Name: Fujitsu Storage Systems ETERNUS DX200 S3	
Metric	**Reported Result**
SPC-1 IOPS™	200,500.95
SPC-1 Price-Performance	$0.77/SPC-1 IOPS™
Total ASU Capacity	3,280.000 GB
Data Protection Level	Protected 2 *(Mirroring)*
Total Price (including three-year maintenance)	$154,005.32
Currency Used	U.S. Dollars
Target Country for availability, sales and support	USA

The following SPC Benchmark 1™ Remote Audit requirements were reviewed and found compliant with 1.14 of the SPC Benchmark 1™ specification:

- A Letter of Good Faith, signed by a senior executive.
- The following Data Repository storage items were verified by information supplied by Fujitsu Limited:
  - ✓ Physical Storage Capacity and requirements.
  - ✓ Configured Storage Capacity and requirements.
  - ✓ Addressable Storage Capacity and requirements.
  - ✓ Capacity of each Logical Volume and requirements.
  - ✓ Capacity of each Application Storage Unit (ASU) and requirements.
- The total Application Storage Unit (ASU) Capacity was filled with random data, using an auditor approved tool, prior to execution of the SPC-1 Tests.

Storage Performance Council
643 Bair Island Road, Suite 103
Redwood City, CA 94062
AuditService@storageperformance.org
650.556.9384

AUDIT CERTIFICATION *(CONT.)*



Fujitsu Storage Systems ETERNUS DX200 S3 Page 2
SPC-1 Audit Certification

- An appropriate diagram of the Benchmark Configuration (BC)/Tested Storage Configuration (TSC).
- Listings and commands to configure the Benchmark Configuration/Tested Storage Configuration, including customer tunable parameters that were changed from default values.
- SPC-1 Workload Generator commands and parameters used for the audited SPC Test Runs.
- The following Host System requirements were verified by information supplied by Fujitsu Limited:
  - ✓ The type of Host Systems including the number of processors and main memory.
  - ✓ The presence and version number of the SPC-1 Workload Generator on each Host System.
  - ✓ The TSC boundary within each Host System.
- The execution of each Test, Test Phase, and Test Run was found compliant with all of the requirements and constraints of Clauses 4, 5, and 11 of the SPC-1 Benchmark Specification.
- The Test Results Files and resultant Summary Results Files received from Fujitsu Limited for each of following were authentic, accurate, and compliant with all of the requirements and constraints of Clauses 4 and 5 of the SPC-1 Benchmark Specification:
  - ✓ Data Persistence Test
  - ✓ Sustainability Test Phase
  - ✓ IOPS Test Phase
  - ✓ Response Time Ramp Test Phase
  - ✓ Repeatability Test
- There were no differences between the Tested Storage Configuration and Priced Storage Configuration.
- The submitted pricing information met all of the requirements and constraints of Clause 8 of the SPC-1 Benchmark Specification.
- The Full Disclosure Report (FDR) met all of the requirements in Clause 9 of the SPC-1 Benchmark Specification.
- This successfully audited SPC measurement is not subject to an SPC Confidential Review.

Audit Notes:

There were no audit notes or exceptions.

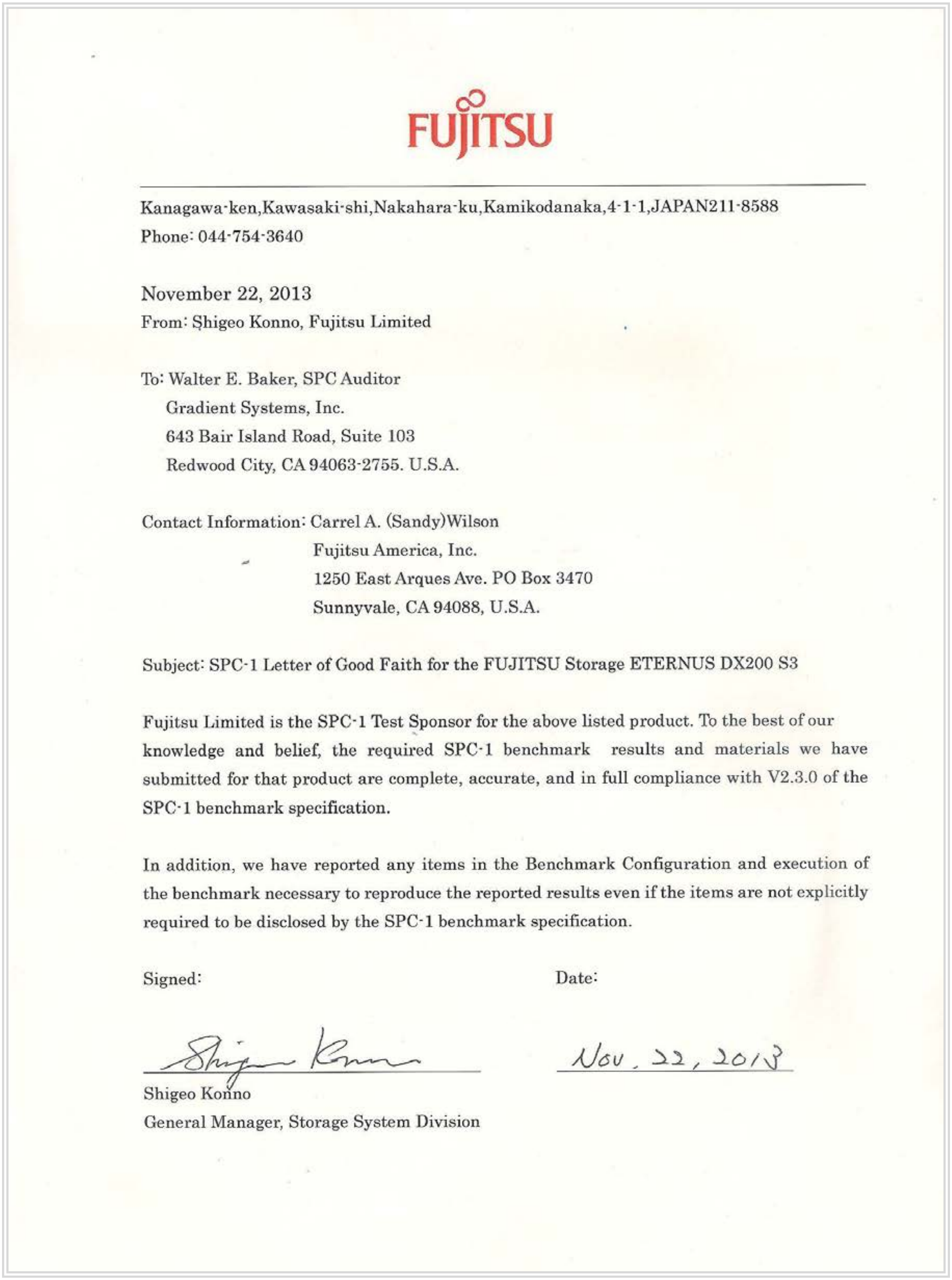
Respectfully,

Walter E. Baker

Walter E. Baker
SPC Auditor

Storage Performance Council
643 Bair Island Road, Suite 103
Redwood City, CA 94062
AuditService@storageperformance.org
650.556.9384

LETTER OF GOOD FAITH



FUJITSU

Kanagawa-ken,Kawasaki-shi,Nakahara-ku,Kamikodanaka,4-1-1,JAPAN211-8588
Phone: 044-754-3640

November 22, 2013
From: Shigeo Konno, Fujitsu Limited

To: Walter E. Baker, SPC Auditor
Gradient Systems, Inc.
643 Bair Island Road, Suite 103
Redwood City, CA 94063-2755. U.S.A.

Contact Information: Carrel A. (Sandy)Wilson
Fujitsu America, Inc.
1250 East Arques Ave. PO Box 3470
Sunnyvale, CA 94088, U.S.A.

Subject: SPC-1 Letter of Good Faith for the FUJITSU Storage ETERNUS DX200 S3

Fujitsu Limited is the SPC-1 Test Sponsor for the above listed product. To the best of our knowledge and belief, the required SPC-1 benchmark results and materials we have submitted for that product are complete, accurate, and in full compliance with V2.3.0 of the SPC-1 benchmark specification.

In addition, we have reported any items in the Benchmark Configuration and execution of the benchmark necessary to reproduce the reported results even if the items are not explicitly required to be disclosed by the SPC-1 benchmark specification.

Signed: [illegible]

Date: Nov. 22, 2013

Shigeo Konno
General Manager, Storage System Division

EXECUTIVE SUMMARY

Test Sponsor and Contact Information

Test Sponsor and Contact Information	
Test Sponsor Primary Contact	Fujitsu Limited – http://www.fujitsu.com/services/computing/storage/ Fujitsu America, Inc. C.A. (Sandy) Wilson swilson@us.fujitsu.com 1250 East Arques Ave PO Box 3470 Sunnyvale, CA 94088-3470 Phone: (916) 434-8593
Test Sponsor Alternate Contact	Fujitsu Limited – http://www.fujitsu.com/services/computing/storage/ Fujitsu America, Inc. Kun Katsumata kkatsumata@us.fujitsu.com 1250 East Arques Ave PO Box 3470 Sunnyvale, CA 94088-3470 Phone: (408) 746-6415
Test Sponsor Alternate Contact	Fujitsu Limited http://www.fujitsu.com/services/computing/storage/ Shigeo Konno konno.shigeo@jp.fujitsu.com 1-1 Kamikodanaka 4-chome, Nakahara-ku, Kawasaki-shi, Kanagawa-ken 211-8588, Japan Phone: (044) 754-3632 FAX: (044) 754-3719
Auditor	Storage Performance Council – http://www.storageperformance.org Walter E. Baker – AuditService@StoragePerformance.org 643 Bair Island Road, Suite 103 Redwood City, CA 94063 Phone: (650) 556-9384 FAX: (650) 556-9385

Revision Information and Key Dates

Revision Information and Key Dates	
SPC-1 Specification revision number	V1.14
SPC-1 Workload Generator revision number	V2.3.0
Date Results were first used publicly	December 30, 2013
Date the FDR was submitted to the SPC	December 30, 2013
Date the Priced Storage Configuration is available for shipment to customers	currently available
Date the TSC completed audit certification	December 20, 2013

Tested Storage Product (TSP) Description

The scalable and unified Fujitsu Storage ETERNUS DX200 S3 delivers enterprise-class functionality to small and medium-sized companies and subsidiaries with an excellent price/performance ratio. It is the perfect solution when consolidating data for server virtualization, e-mail, databases and business applications as well as centralized file services. Simple, intuitive system management, highly flexible network connectivity and the option of field upgrades to higher system classes significantly reduce operational and migration costs. The ETERNUS DX family architecture lets customers benefit from software options, such as thin provisioning, automatic storage tiering and quality of service management even as early as the entry-level class. This all contributes to better business support.

Summary of Results

SPC-1 Reported Data	
Tested Storage Product (TSP) Name: Fujitsu Storage Systems ETERNUS DX200 S3	
Metric	**Reported Result**
SPC-1 IOPS™	200,500.95
SPC-1 Price-Performance™	$0.77/SPC-1 IOPS™
Total ASU Capacity	3,280.000 GB
Data Protection Level	Protected 2 *(Mirroring)*
Total Price	$154,005.32
Currency Used	U.S. Dollars
Target Country for availability, sales and support	USA

SPC-1 IOPS™ represents the maximum I/O Request Throughput at the 100% load point.

SPC-1 Price-Performance™ is the ratio of **Total Price** to **SPC-1 IOPS™**.

Total ASU (Application Storage Unit) **Capacity** represents the total storage capacity available to be read and written in the course of executing the SPC-1 benchmark.

A **Data Protection Level** of **Protected 2** using ***Mirroring,*** which configures two or more identical copies of user data.

> ***Protected 2:*** *The single point of failure of any* ***component*** *in the configuration will not result in permanent loss of access to or integrity of the SPC-1 Data Repository.*

Total Price includes the cost of the Priced Storage Configuration plus three years of hardware maintenance and software support as detailed on page 18.

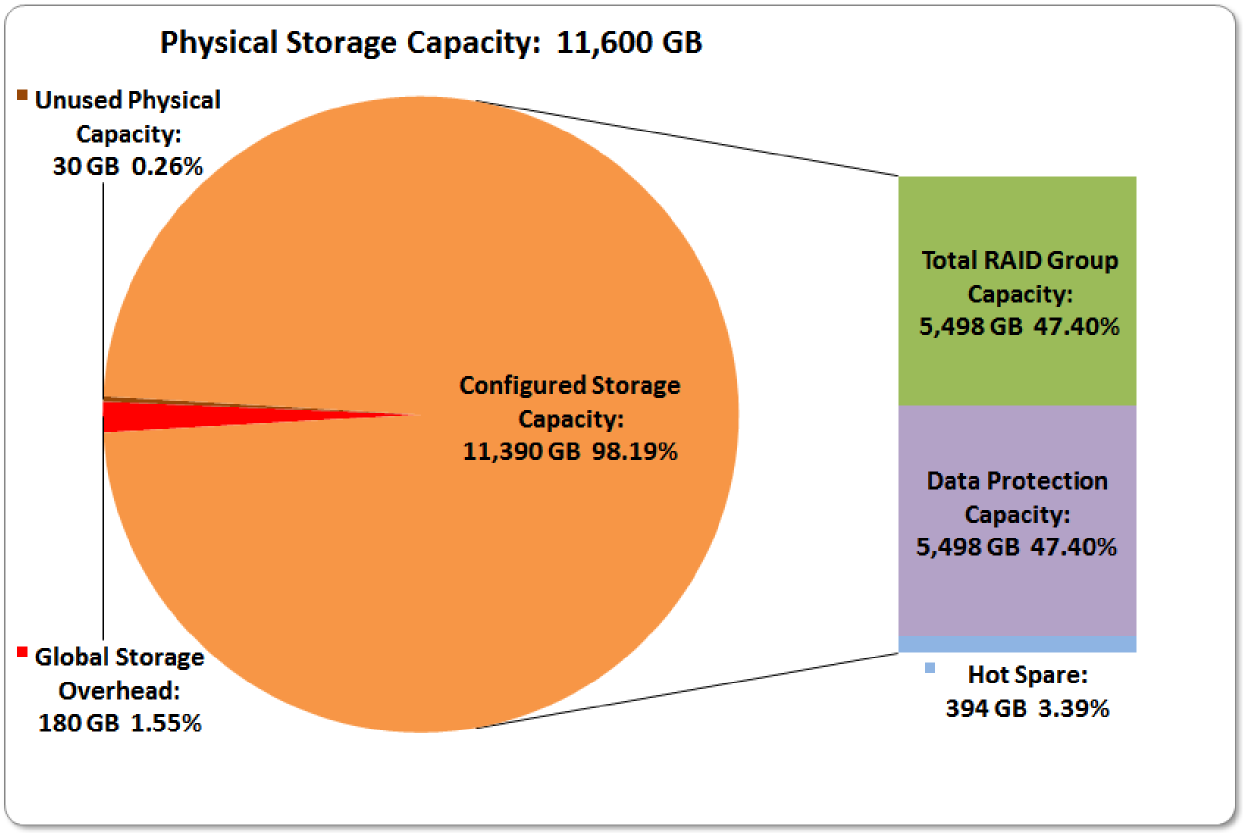
Currency Used is formal name for the currency used in calculating the **Total Price** and **SPC-1 Price-Performance™**. That currency may be the local currency of the **Target Country** or the currency of a difference country *(non-local currency).*

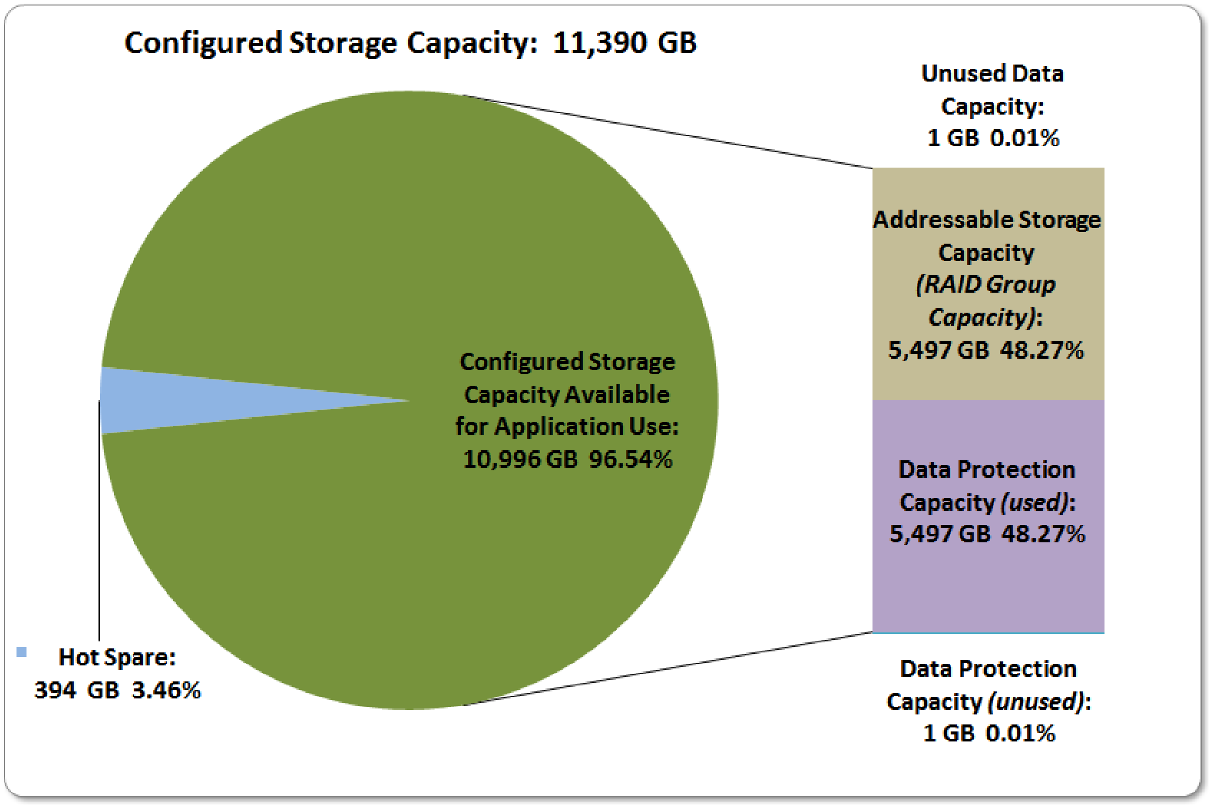
The **Target Country** is the country in which the Priced Storage Configuration is available for sale and in which the required hardware maintenance and software support is provided either directly from the Test Sponsor or indirectly via a third-party supplier.

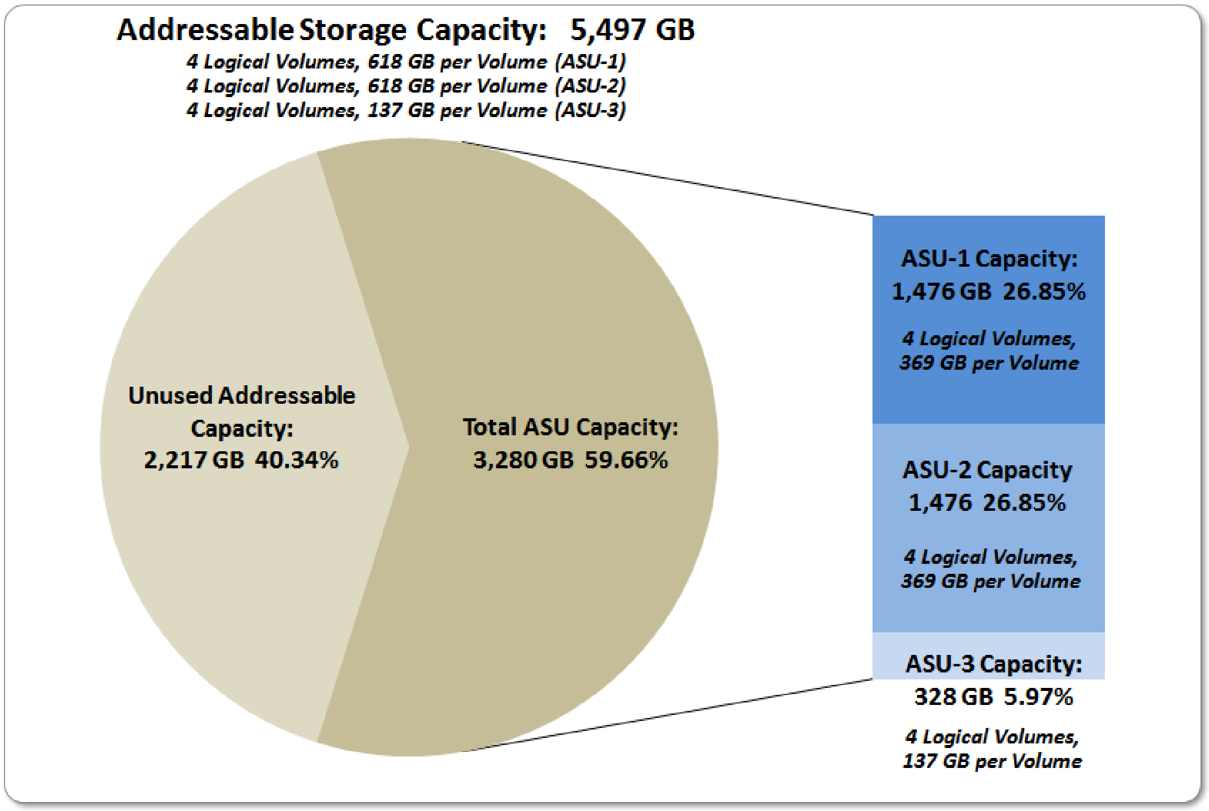
Storage Capacities, Relationships, and Utilization

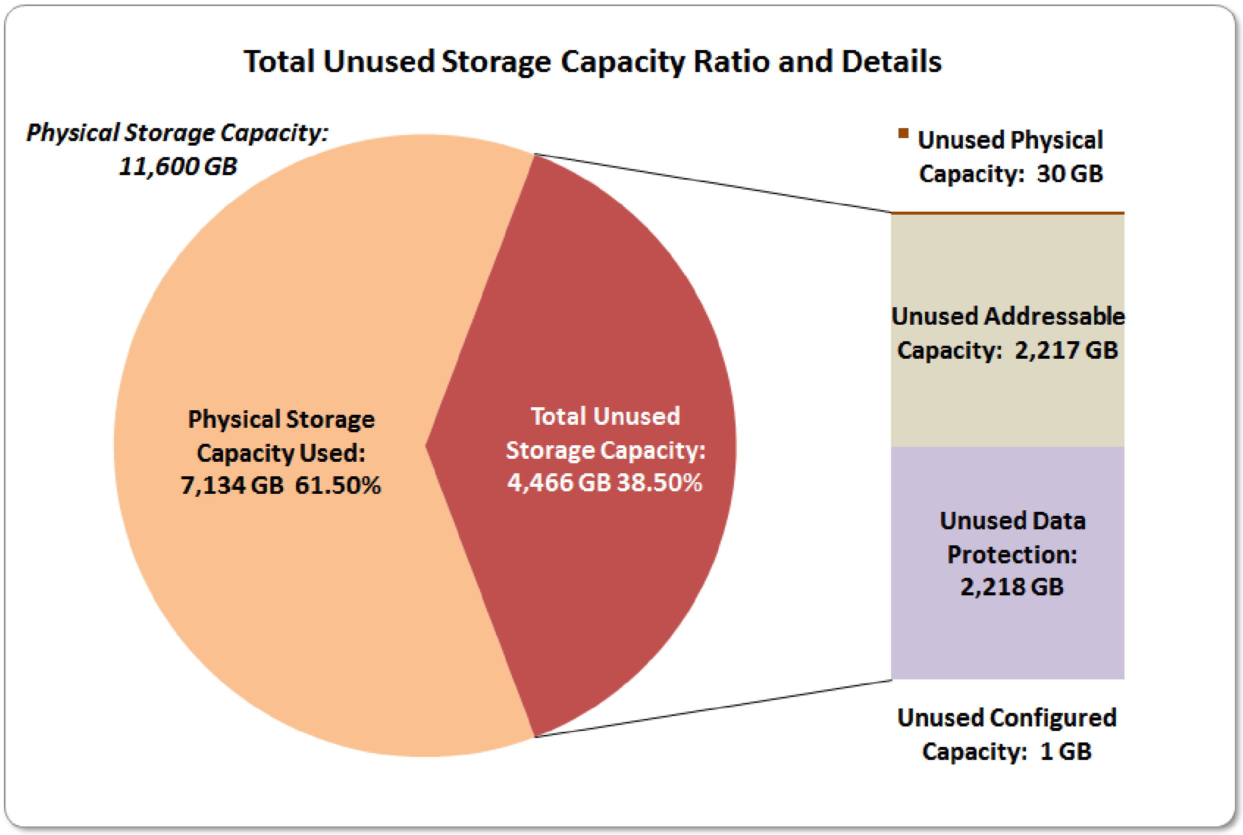
The following four charts and table document the various storage capacities, used in this benchmark, and their relationships, as well as the storage utilization values required to be reported.

The capacity values in each of the following four charts are listed as integer values, for readability, rather than the decimal values listed elsewhere in this document.




Configured Storage Capacity: 11,390 GB
Unused Data Capacity: 1 GB 0.01%
Addressable Storage Capacity *(RAID Group Capacity)*: 5,497 GB 48.27%
Configured Storage Capacity Available for Application Use: 10,996 GB 96.54%
Data Protection Capacity *(used)*: 5,497 GB 48.27%
Hot Spare: 394 GB 3.46%
Data Protection Capacity *(unused)*: 1 GB 0.01%


Addressable Storage Capacity: 5,497 GB
4 Logical Volumes, 618 GB per Volume (ASU-1)
4 Logical Volumes, 618 GB per Volume (ASU-2)
4 Logical Volumes, 137 GB per Volume (ASU-3)
ASU-1 Capacity: 1,476 GB 26.85%
4 Logical Volumes, 369 GB per Volume
Unused Addressable Capacity: 2,217 GB 40.34%
Total ASU Capacity: 3,280 GB 59.66%
ASU-2 Capacity 1,476 26.85%
4 Logical Volumes, 369 GB per Volume
ASU-3 Capacity: 328 GB 5.97%
4 Logical Volumes, 137 GB per Volume



SPC-1 Storage Capacity Utilization	
Application Utilization	28.28%
Protected Application Utilization	56.55%
Unused Storage Ratio	38.50%

Application Utilization: Total ASU Capacity *(3,280.000 GB)* divided by Physical Storage Capacity *(11,600.000 GB).*

Protected Application Utilization: Total ASU Capacity *(3,280.000 GB)* plus total Data Protection Capacity *(5,498.095 GB)* minus unused Data Protection Capacity *(2,218,095 GB)* divided by Physical Storage Capacity *(11,600.000 GB).*

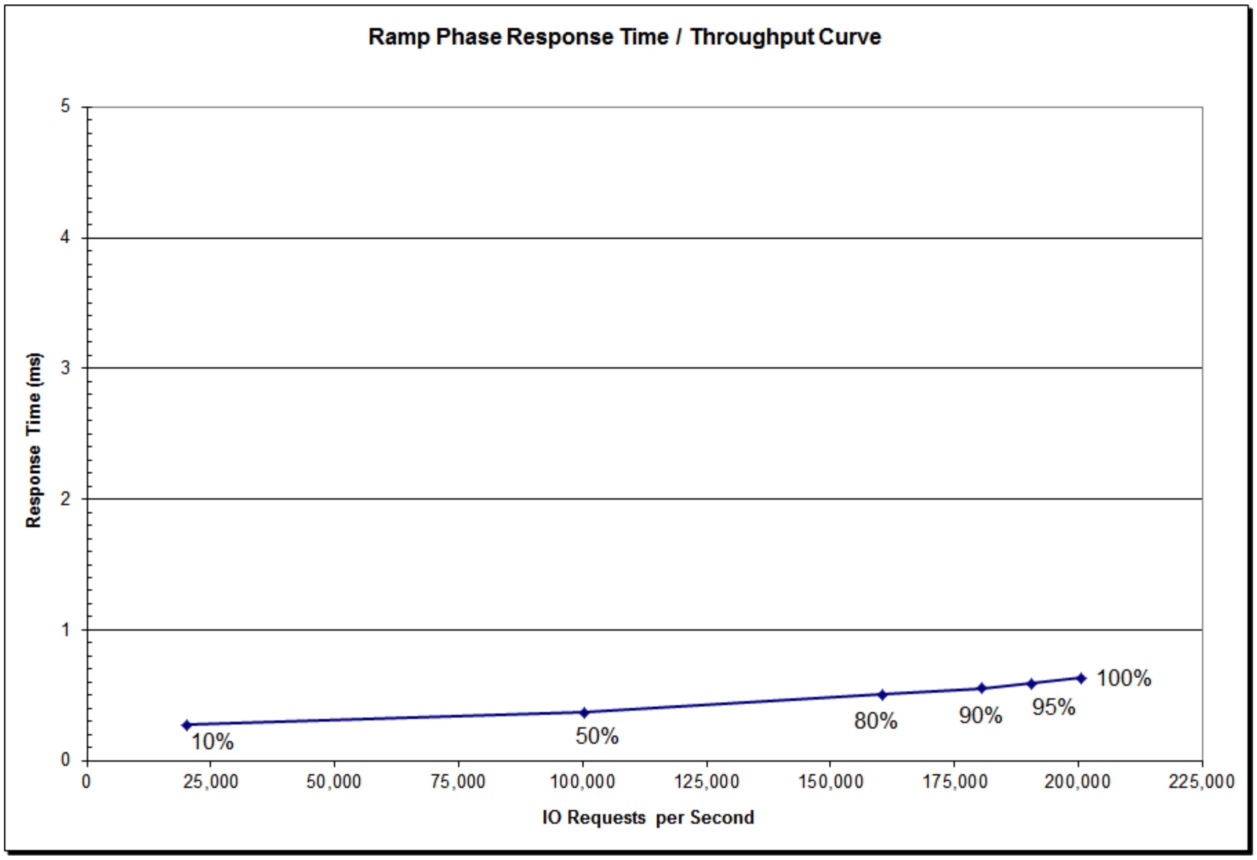
Unused Storage Ratio: Total Unused Capacity *(4,466.255 GB)* divided by Physical Storage Capacity *(11,600.000 GB)* and may not exceed 45%.

Detailed information for the various storage capacities and utilizations is available on pages 25-26.

Response Time – Throughput Curve

The Response Time-Throughput Curve illustrates the Average Response Time (milliseconds) and I/O Request Throughput at 100%, 95%, 90%, 80%, 50%, and 10% of the workload level used to generate the SPC-1 IOPS™ metric.

The Average Response Time measured at any of the above load points cannot exceed 30 milliseconds or the benchmark measurement is invalid.



Response Time – Throughput Data

	10% Load	50% Load	80% Load	90% Load	95% Load	100% Load
I/O Request Throughput	20,053.34	100,240.80	160,412.21	180,467.93	190,491.94	200,453.37
Average Response Time (ms):						
All ASUs	0.27	0.37	0.50	0.55	0.59	0.63
ASU-1	0.29	0.42	0.59	0.65	0.69	0.74
ASU-2	0.29	0.41	0.55	0.59	0.65	0.67
ASU-3	0.23	0.24	0.30	0.33	0.36	0.40
Reads	0.36	0.59	0.83	0.91	0.95	1.02
Writes	0.21	0.23	0.29	0.32	0.36	0.38

Priced Storage Configuration Pricing

Product ID	Product Name	Qty	Unit List Price	Extended LP	Discount %	Discounted Price
ET203AU	DX200 S3 Base System Rackmount (AC200V, 2RU) (2.5" type)	1	$3,579.00	$3,579.00	30%	$2,505.30
ETFEADU	Drive Enclosure - DX100/200 Rackmount (AC200V, 2RU) (2.5" HDD, Dual IOM type)	1	$4,275.00	$4,275.00	30%	$2,992.50
ETFM42U	Cache Memory - DX200 S3 8GB per CM	2	$1,000.00	$2,000.00	30%	$1,400.00
ETFCH2F	FC Host Interface, 2 ports DX200 S3 CM & CA#0 (4/8/16Gbps, Host/Remote Connect)	2	$8,806.00	$17,612.00	30%	$12,328.40
ETFHH2	FC Host Interface, 2 ports - CA#1 (4/8/16Gbps, Host/Remote Connect)	2	$2,000.00	$4,000.00	30%	$2,800.00
ETFSA4	MLC-SSD 400GB 2.5" Drive x1 SAS for DX100/200 S3	29	$4,422.00	$128,238.00	30%	$89,766.60
QLE2562	QLogic 8Gbps Dual Port Fibre Channel Host Bus Adapter	8	$2,597.85	$20,782.80	10%	$18,704.52
ETFKC05U	AC Power Cords (125V - IEC320-C14, 0.5m)	2	$80.00	$160.00	30%	$112.00
61-343827-003	Fibre Channel Host IF Cable LC/LC - 3m	8	$132.00	$1,056.00	30%	$739.20

	(Provide 24 hour per day / 7days per week 4 hour response maintenance for 36 months)					
ETD200-W025360-ADE	Warranty Service, 36 months Standard, 9x5 phone, NBD response	1	$0.00	$0.00		$0.00
ETD200-U004361-ADE	Warranty Uplift, 36 months Enhanced plus, 24x7 4hr Onsite	1	$28,321.00	$28,321.00	20%	$22,656.80
	SFPs are included.			**Total:**		**$154,005.32**

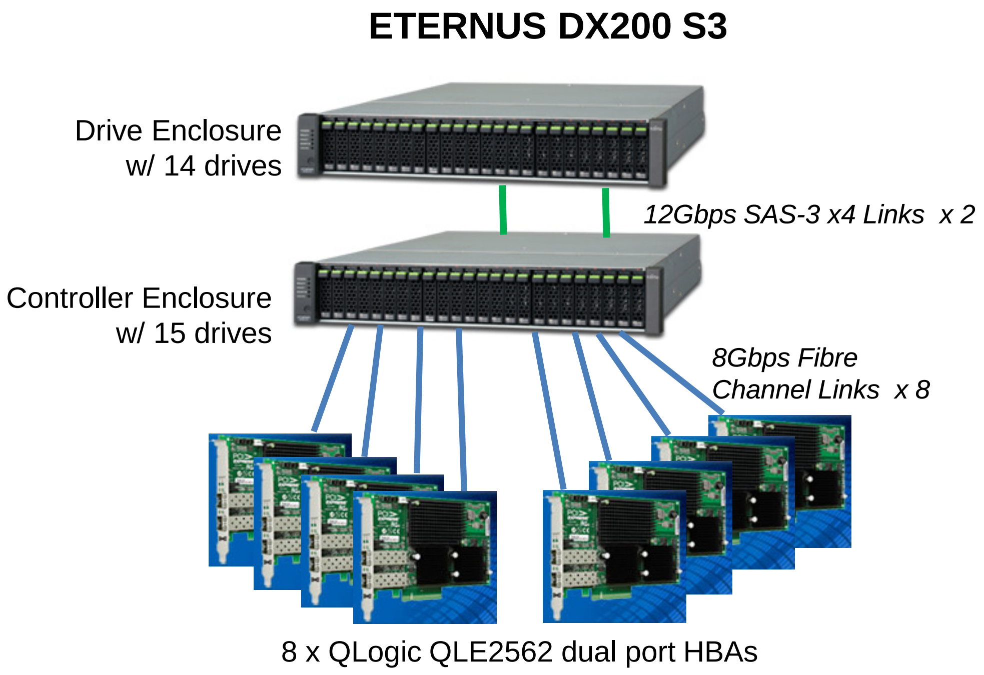
The above pricing includes hardware maintenance and software support for three years, 7 days per week, 24 hours per day. The hardware maintenance and software support provides the following:

- Acknowledgement of new and existing problems with four (4) hours.
- Onsite presence of a qualified maintenance engineer or provision of a customer replaceable part within four (4) hours of the above acknowledgement for any hardware failure that results in an inoperative Price Storage Configuration that can be remedied by the repair or replacement of a Priced Storage Configuration component.

Differences between the Tested Storage Configuration (TSC) and Priced Storage Configuration

There were no differences between the TSC and the Priced Storage Configuration.

Priced Storage Configuration Diagram


ETERNUS DX200 S3
Drive Enclosure w/ 14 drives
12Gbps SAS-3 x4 Links x 2
Controller Enclosure w/ 15 drives
8Gbps Fibre Channel Links x 8
8 x QLogic QLE2562 dual port HBAs

Priced Storage Configuration Components

Priced Storage Configuration
8 – QLogic QLE2562 dual-port HBAs
Fujitsu Storage Systems ETERNUS DX200 S3 1 – Controller Enclosure Module with 2 – Control Modules (CM) each with 8 GB cache (16 GB total) 2 – Channel Adapters (CA) each with 2 – 16 Gbps FC host ports *(4 ports per CA, 8 ports total and used)* 1 – SAS Expander Drive Interface with QSFP 12 Gbps SAS-3 *(2 – SAS-3 x4 links total and used)* 24 – Hot Swap drive slots 15 – 400 GB MLC SSD storage devices
1 – Drive Enclosure Module with 24 – Hot Swap drive slots 14 – 400 GB MLC SSD storage devices 2 – SAS Interface/Expander Modules with QSFP 12 Gbps SAS-3

In each of the following sections of this document, the appropriate Full Disclosure Report requirement, from the SPC-1 benchmark specification, is stated in italics followed by the information to fulfill the stated requirement.

CONFIGURATION INFORMATION

Benchmark Configuration (BC)/Tested Storage Configuration (TSC) Diagram

Clause 9.4.3.4.1

A one page Benchmark Configuration (BC)/Tested Storage Configuration (TSC) diagram shall be included in the FDR…

The Benchmark Configuration (BC)/Tested Storage Configuration (TSC) is illustrated on page 21 (*Benchmark Configuration/Tested Storage Configuration Diagram*).

Storage Network Configuration

Clause 9.4.3.4.1

…

5. *If the TSC contains network storage, the diagram will include the network configuration. If a single diagram is not sufficient to illustrate both the Benchmark Configuration and network configuration in sufficient detail, the Benchmark Configuration diagram will include a high-level network illustration as shown in Figure 9-8. In that case, a separate, detailed network configuration diagram will also be included as described in Clause 9.4.3.4.2.*

Clause 9.4.3.4.2

If a storage network was configured as a part of the Tested Storage Configuration and the Benchmark Configuration diagram described in Clause 9.4.3.4.1 contains a high-level illustration of the network configuration, the Executive Summary will contain a one page topology diagram of the storage network as illustrated in Figure 9-9.

The Benchmark Configuration (BC)/Tested Storage Configuration (TSC) was configured with local storage and, as such, did not employ a storage network.

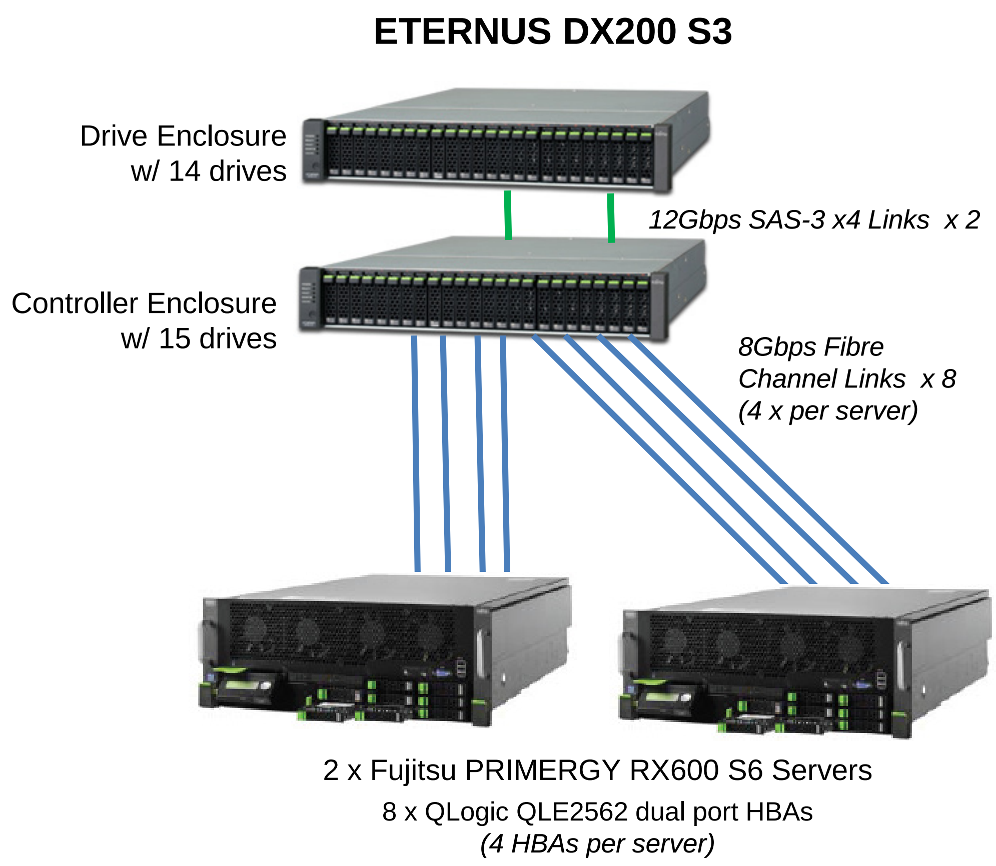
Host System(s) and Tested Storage Configuration (TSC) Table of Components

Clause 9.4.3.4.3

The FDR will contain a table that lists the major components of each Host System and the Tested Storage Configuration (TSC).

The Host System(s) and TSC table of components may be found on page 22 (*Host Systems and Tested Storage Configuration Components*).

Benchmark Configuration/Tested Storage Configuration Diagram


ETERNUS DX200 S3
Drive Enclosure w/ 14 drives
12Gbps SAS-3 x4 Links x 2
Controller Enclosure w/ 15 drives
8Gbps Fibre Channel Links x 8 (4 x per server)
2 x Fujitsu PRIMERGY RX600 S6 Servers
8 x QLogic QLE2562 dual port HBAs
(4 HBAs per server)

Host Systems and Tested Storage Configuration Components

Host Systems
2 – Fujitsu PRIMERGY RX600 S6 Servers, each with: 4 – Intel Xeon 2.67 GHz processor E7-8837 each with 8 cores, 24 MB cache 64 GB main memory Red Hat Enterprise Linux Server release 6.2 PCI-Express 2.0
Tested Storage Configuration (TSC)
8 – QLogic QLE2562 dual-port HBAs *(4 HBAs per server)*
Fujitsu Storage Systems ETERNUS DX200 S3 1 – Controller Enclosure Module with 2 – Control Modules (CM) each with 8 GB cache (16 GB total) 2 – Channel Adapters (CA) each with 2 – 16 Gbps FC host ports *(4 ports per CA, 8 ports total and used)* 1 – SAS Expander Drive Interface with QSFP 12 Gbps SAS-3 *(2 – SAS-3 x4 links total and used)* 24 – Hot Swap drive slots 15 – 400 GB MLC SSD storage devices
1 – Drive Enclosure Module with 24 – Hot Swap drive slots 14 – 400 GB MLC SSD storage devices 2 – SAS Interface/Expander Modules with QSFP 12 Gbps SAS-3

Customer Tunable Parameters and Options

Clause 9.4.3.5.1

All Benchmark Configuration (BC) components with customer tunable parameter and options that have been altered from their default values must be listed in the FDR. The FDR entry for each of those components must include both the name of the component and the altered value of the parameter or option. If the parameter name is not self-explanatory to a knowledgeable practitioner, a brief description of the parameter's use must also be included in the FDR entry.

Appendix B: Customer Tunable Parameters and Options on page 66 contains the customer tunable parameters and options that have been altered from their default values for this benchmark.

Tested Storage Configuration (TSC) Description

Clause 9.4.3.5.2

The FDR must include sufficient information to recreate the logical representation of the TSC. In addition to customer tunable parameters and options (Clause 4.2.4.5.3), that information must include, at a minimum:

- *A diagram and/or description of the following:*
  - *All physical components that comprise the TSC. Those components are also illustrated in the BC Configuration Diagram in Clause 9.2.4.4.1 and/or the Storage Network Configuration Diagram in Clause 9.2.4.4.2.*
  - *The logical representation of the TSC, configured from the above components that will be presented to the Workload Generator.*
- *Listings of scripts used to create the logical representation of the TSC.*
- *If scripts were not used, a description of the process used with sufficient detail to recreate the logical representation of the TSC.*

Appendix C: Tested Storage Configuration (TSC) Creation on page 67 contains the detailed information that describes how to create and configure the logical TSC.

SPC-1 Workload Generator Storage Configuration

Clause 9.4.3.5.3

The FDR must include all SPC-1 Workload Generator storage configuration commands and parameters.

The SPC-1 Workload Generator storage configuration commands and parameters for this measurement appear in Appendix D: SPC-1 Workload Generator Storage Commands and Parameters on page 71.

ASU Pre-Fill

Clause 5.3.3

Each of the three SPC-1 ASUs (ASU-1, ASU-2 and ASU-3) is required to be completely filled with specified content prior to the execution of audited SPC-1 Tests. The content is required to consist of random data pattern such as that produced by an SPC recommended tool.

The configuration file used to complete the required ASU pre-fill appears in Appendix D: SPC-1 Workload Generator Storage Commands and Parameters on page 71.

SPC-1 DATA REPOSITORY

This portion of the Full Disclosure Report presents the detailed information that fully documents the various SPC-1 storage capacities and mappings used in the Tested Storage Configuration. SPC-1 Data Repository Definitions on page 62 contains definitions of terms specific to the SPC-1 Data Repository.

Storage Capacities and Relationships

Clause 9.4.3.6.1

Two tables and four charts documenting the storage capacities and relationships of the SPC-1 Storage Hierarchy (Clause 2.1) shall be included in the FDR. … The capacity value in each chart may be listed as an integer value, for readability, rather than the decimal value listed in the table below.

SPC-1 Storage Capacities

The Physical Storage Capacity consisted of 11,600.000 GB distributed over 29 solid state storage devices, each with a formatted capacity of 400.000 GB. There was 30.065 GB (0.26%) of Unused Storage within the Physical Storage Capacity. Global Storage Overhead consisted of 179.951 GB (1.55%) of the Physical Storage Capacity. There was 0.709 GB (0.01%) of Unused Storage within the Configured Storage Capacity. The Total ASU Capacity utilized 59.66% of the Addressable Storage Capacity resulting in 2,217.386 GB (40.34%) of Unused Storage within the Addressable Storage Capacity. The Data Protection *(mirroring)* capacity was 5,498.095 GB of which 3,280.000 GB was utilized. The total Unused Storage capacity was 4,466.255 GB.

Note: The configured Storage Devices may include additional storage capacity reserved for system overhead, which is not accessible for application use. That storage capacity may not be included in the value presented for Physical Storage Capacity.

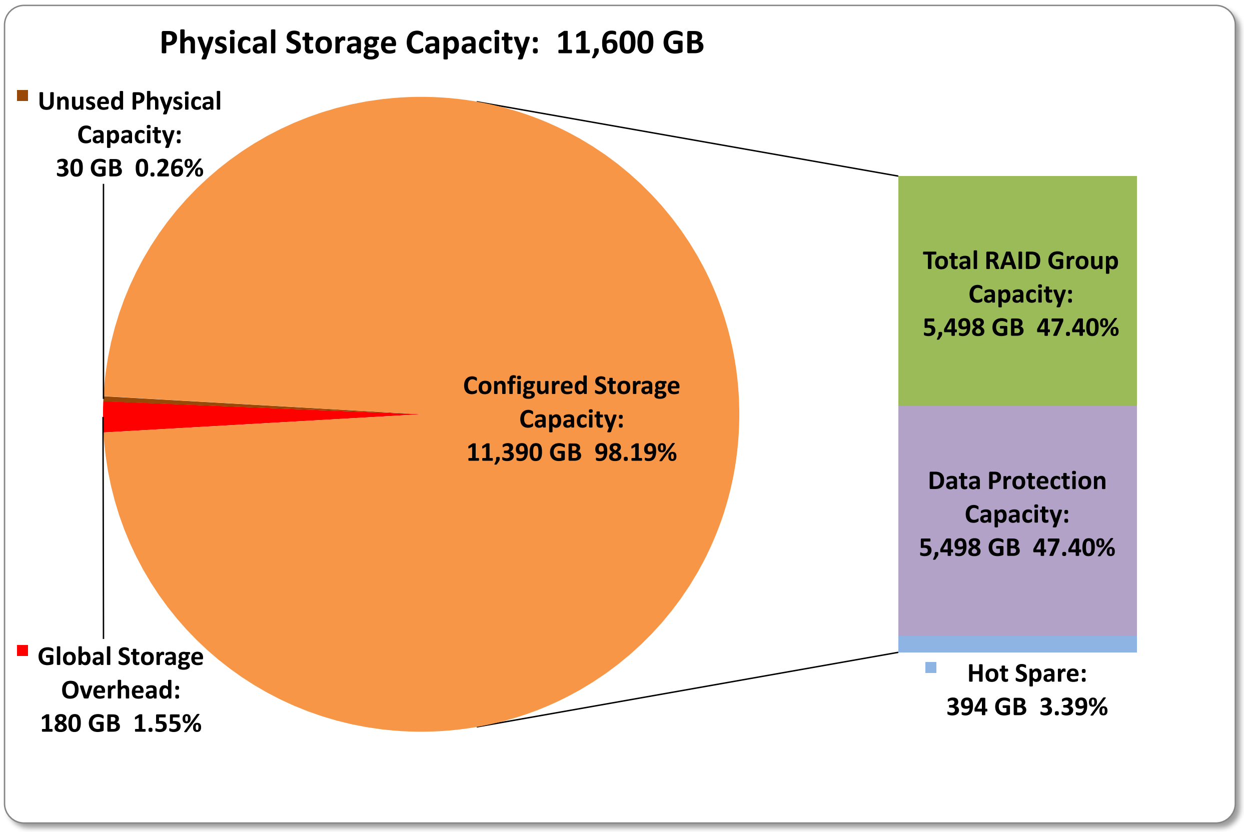
SPC-1 Storage Capacities

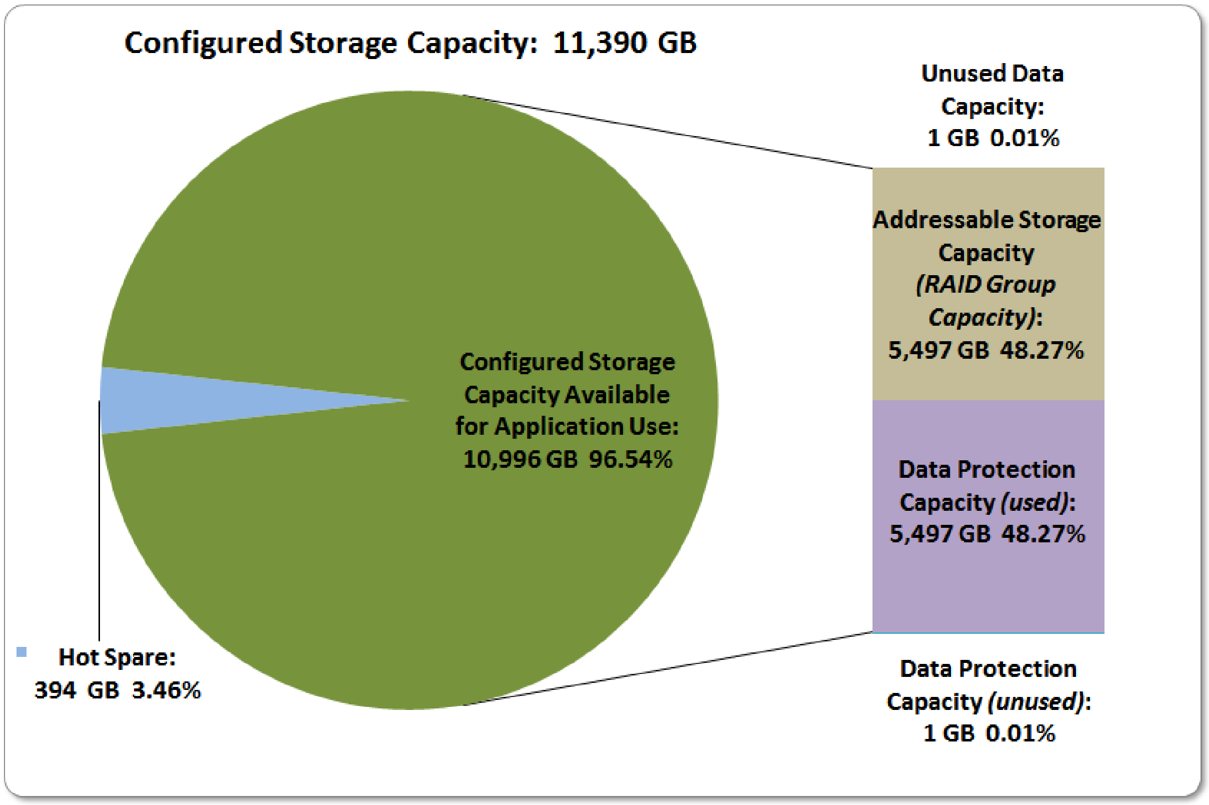
Storage Hierarchy Component	Units	Capacity
Total ASU Capacity	Gigabytes (GB)	3,280.000
Addressable Storage Capacity	Gigabytes (GB)	5,497.386
Configured Storage Capacity	Gigabytes (GB)	11,389.985
Physical Storage Capacity	Gigabytes (GB)	11,600.000
Data Protection *(Mirroring)*	Gigabytes (GB)	5,498.095
Required Storage *(hot spare)*	Gigabytes (GB)	393.795
Global Storage Overhead	Gigabytes (GB)	179.951
Total Unused Storage	Gigabytes (GB)	4,466.255

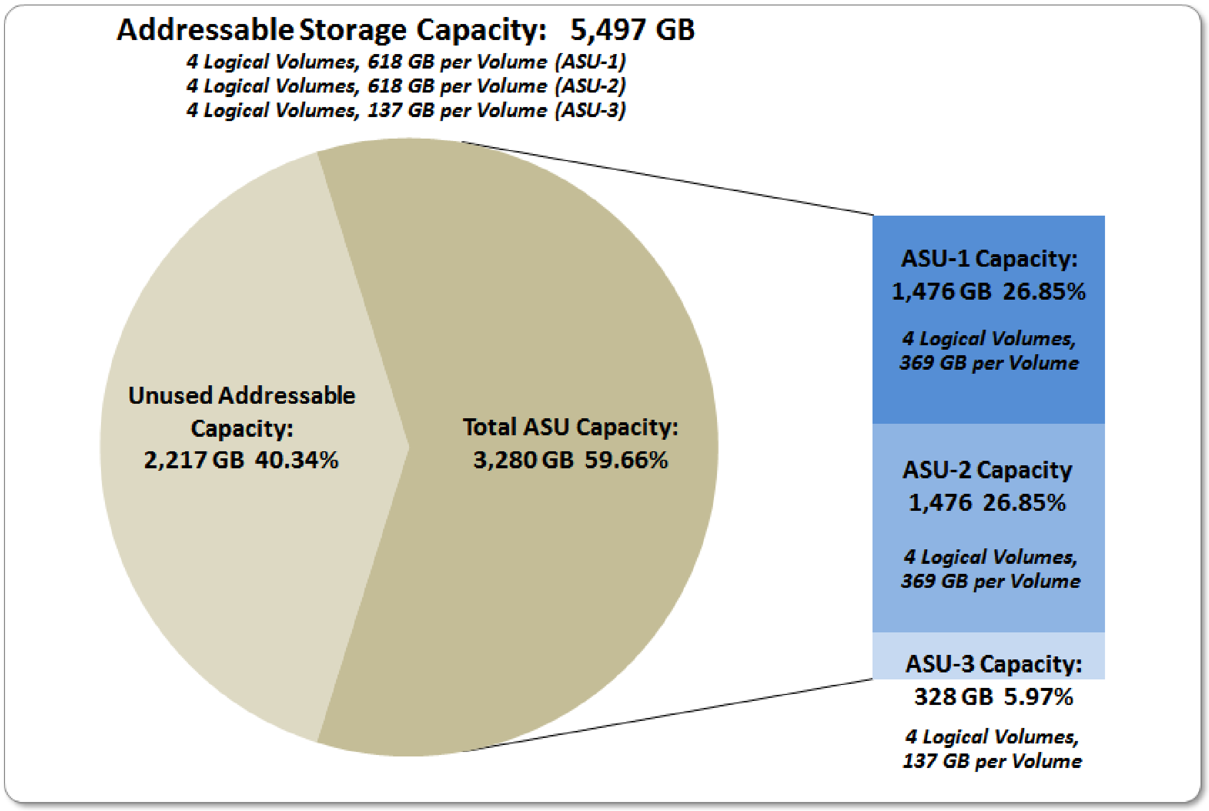
SPC-1 Storage Hierarchy Ratios

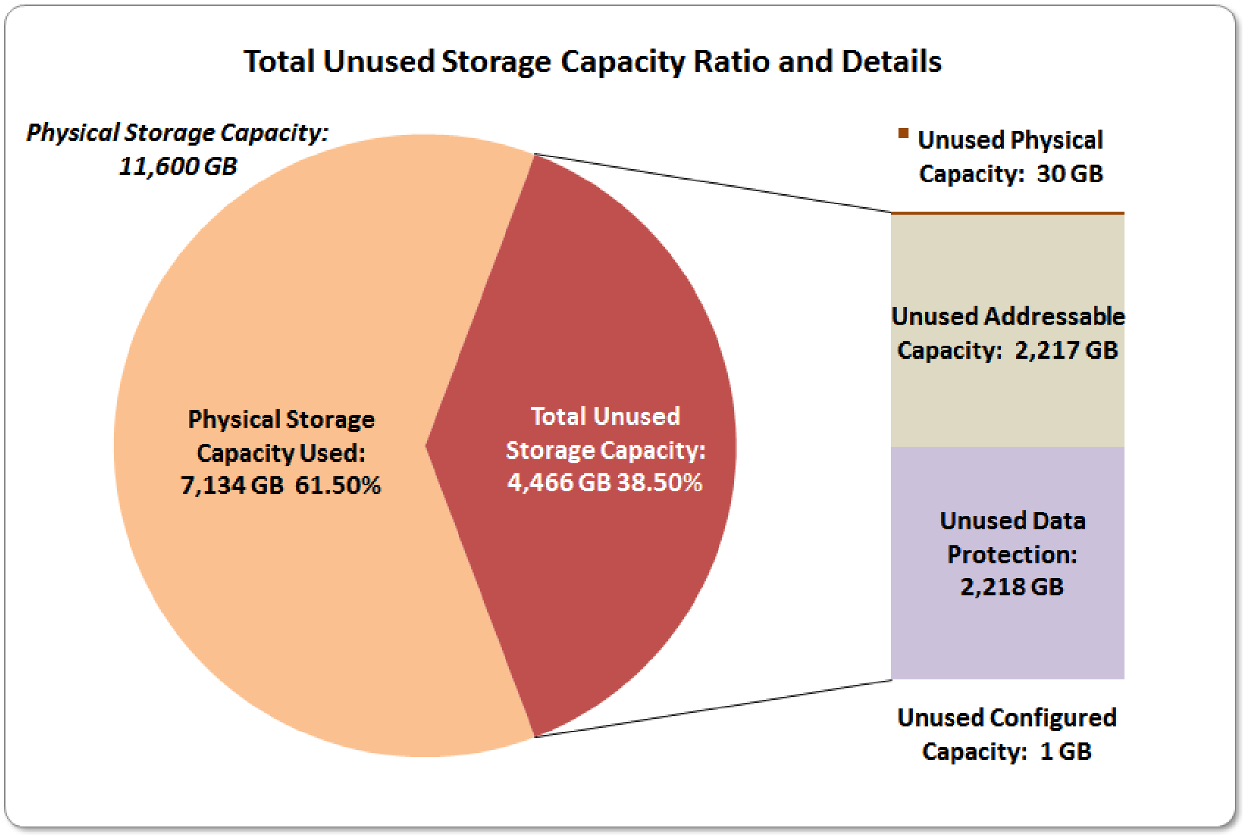
	Addressable Storage Capacity	Configured Storage Capacity	Physical Storage Capacity
Total ASU Capacity	59.66%	28.80%	28.28%
Required for Data Protection *(Mirroring)*		48.27%	47.40%
Addressable Storage Capacity		48.27%	47.39%
Required Storage *(hot spare)*		3.46%	3.39%
Configured Storage Capacity			98.19%
Global Storage Overhead			1.55%
Unused Storage:			
Addressable	40.34%		
Configured		0.01%	
Physical			0.26%

SPC-1 Storage Capacity Charts




Configured Storage Capacity: 11,390 GB
Unused Data Capacity: 1 GB 0.01%
Addressable Storage Capacity *(RAID Group Capacity)*: 5,497 GB 48.27%
Configured Storage Capacity Available for Application Use: 10,996 GB 96.54%
Data Protection Capacity *(used)*: 5,497 GB 48.27%
Hot Spare: 394 GB 3.46%
Data Protection Capacity *(unused)*: 1 GB 0.01%


Addressable Storage Capacity: 5,497 GB
4 Logical Volumes, 618 GB per Volume (ASU-1)
4 Logical Volumes, 618 GB per Volume (ASU-2)
4 Logical Volumes, 137 GB per Volume (ASU-3)
ASU-1 Capacity: 1,476 GB 26.85%
4 Logical Volumes, 369 GB per Volume
Unused Addressable Capacity: 2,217 GB 40.34%
Total ASU Capacity: 3,280 GB 59.66%
ASU-2 Capacity 1,476 26.85%
4 Logical Volumes, 369 GB per Volume
ASU-3 Capacity: 328 GB 5.97%
4 Logical Volumes, 137 GB per Volume



Storage Capacity Utilization

Clause 9.4.3.6.2

The FDR will include a table illustrating the storage capacity utilization values defined for Application Utilization (Clause 2.8.1), Protected Application Utilization (Clause 2.8.2), and Unused Storage Ratio (Clause 2.8.3).

Clause 2.8.1

Application Utilization is defined as Total ASU Capacity divided by Physical Storage Capacity.

Clause 2.8.2

Protected Application Utilization is defined as (Total ASU Capacity plus total Data Protection Capacity minus unused Data Protection Capacity) divided by Physical Storage Capacity.

Clause 2.8.3

Unused Storage Ratio is defined as Total Unused Capacity divided by Physical Storage Capacity and may not exceed 45%.

SPC-1 Storage Capacity Utilization	
Application Utilization	28.28%
Protected Application Utilization	56.55%
Unused Storage Ratio	38.50%

Logical Volume Capacity and ASU Mapping

Clause 9.4.3.6.3

A table illustrating the capacity of each ASU and the mapping of Logical Volumes to ASUs shall be provided in the FDR. … Logical Volumes shall be sequenced in the table from top to bottom per its position in the contiguous address space of each ASU. The capacity of each Logical Volume shall be stated. … In conjunction with this table, the Test Sponsor shall provide a complete description of the type of data protection (see Clause 2.4.5) used on each Logical Volume.

Logical Volume Capacity and Mapping

ASU-1 (1,476.000 GB)	ASU-2 (1,476.000 GB)	ASU-3 (328.000 GB)
4 Logical Volumes 618.476 GB per Logical Volume (369.000 GB used per Logical Volume)	4 Logical Volumes 618.476 GB per Logical Volume (369.000 GB used per Logical Volume)	4 Logical Volumes 137.396 GB per Logical Volume (82.000 GB used per Logical Volume)

The Data Protection Level used for all Logical Volumes was **Protected 2** using ***Mirroring*** as described on page 13. See "ASU Configuration" in the **IOPS Test Results File** for more detailed configuration information.

SPC-1 BENCHMARK EXECUTION RESULTS

This portion of the Full Disclosure Report documents the results of the various SPC-1 Tests, Test Phases, and Test Runs. An SPC-1 glossary on page 62 contains definitions of terms specific to the SPC-1 Tests, Test Phases, and Test Runs.

Clause 5.4.3

The Tests must be executed in the following sequence: Primary Metrics, Repeatability, and Data Persistence. That required sequence must be uninterrupted from the start of Primary Metrics to the completion of Persistence Test Run 1. Uninterrupted means the Benchmark Configuration shall not be power cycled, restarted, disturbed, altered, or adjusted during the above measurement sequence. If the required sequence is interrupted other than for the Host System/TSC power cycle between the two Persistence Test Runs, the measurement is invalid.

SPC-1 Tests, Test Phases, and Test Runs

The SPC-1 benchmark consists of the following Tests, Test Phases, and Test Runs:

- **Primary Metrics Test**
  - Sustainability Test Phase and Test Run
  - IOPS Test Phase and Test Run
  - Response Time Ramp Test Phase
    - 95% of IOPS Test Run
    - 90% of IOPS Test Run
    - 80% of IOPS Test Run
    - 50% of IOPS Test Run
    - 10% of IOPS Test Run (LRT)
- **Repeatability Test**
  - Repeatability Test Phase 1
    - 10% of IOPS Test Run (LRT)
    - IOPS Test Run
  - Repeatability Test Phase 2
    - 10% of IOPS Test Run (LRT)
    - IOPS Test Run
- **Data Persistence Test**
  - Data Persistence Test Run 1
  - Data Persistence Test Run 2

Each Test is an atomic unit that must be executed from start to finish before any other Test, Test Phase, or Test Run may be executed.

The results from each Test, Test Phase, and Test Run are listed below along with a more detailed explanation of each component.

"Ramp-Up" Test Runs

Clause 5.3.13

In order to warm-up caches or perform the initial ASU data migration in a multi-tier configuration, a Test Sponsor may perform a series of "Ramp-Up" Test Runs as a substitute for an initial, gradual Ramp-Up.

Clause 5.3.13.3

The "Ramp-Up" Test Runs will immediately precede the Primary Metrics Test as part of the uninterrupted SPC-1 measurement sequence.

Clause 9.4.3.7.1

If a series of "Ramp-Up" Test Runs were included in the SPC-1 measurement sequence, the FDR shall report the duration (ramp-up and measurement interval), BSU level, SPC-1 IOPS and average response time for each "Ramp-Up" Test Run in an appropriate table.

There were no "Ramp-Up" Test Runs executed in this set of benchmark measurements.

Primary Metrics Test – Sustainability Test Phase

Clause 5.4.4.1.1

The Sustainability Test Phase has exactly one Test Run and shall demonstrate the maximum sustainable I/O Request Throughput within at least a continuous eight (8) hour Measurement Interval. This Test Phase also serves to insure that the TSC has reached Steady State prior to reporting the final maximum I/O Request Throughput result (SPC-1 IOPS™).

Clause 5.4.4.1.2

The computed I/O Request Throughput of the Sustainability Test must be within 5% of the reported SPC-1 IOPS™ result.

Clause 5.4.4.1.4

The Average Response Time, as defined in Clause 5.1.1, will be computed and reported for the Sustainability Test Run and cannot exceed 30 milliseconds. If the Average Response time exceeds that 30-milliseconds constraint, the measurement is invalid.

Clause 9.4.3.7.2

For the Sustainability Test Phase the FDR shall contain:

1. *A Data Rate Distribution graph and data table.*
2. *I/O Request Throughput Distribution graph and data table.*
3. *A Response Time Frequency Distribution graph and table.*
4. *An Average Response Time Distribution graph and table.*
5. *The human readable Test Run Results File produced by the Workload Generator (may be included in an appendix).*
6. *A listing or screen image of all input parameters supplied to the Workload Generator (may be included in an appendix).*
7. *The Measured Intensity Multiplier for each I/O stream.*
8. *The variability of the Measured Intensity Multiplier, as defined in Clause 5.3.13.3.*

SPC-1 Workload Generator Input Parameters

The SPC-1 Workload Generator input parameters for the Sustainability, IOPS, Response Time Ramp, Repeatability, and Persistence Test Runs are documented in Appendix E: SPC-1 Workload Generator Input Parameters on Page 76.

Sustainability Test Results File

A link to the test results file generated from the Sustainability Test Run is listed below.

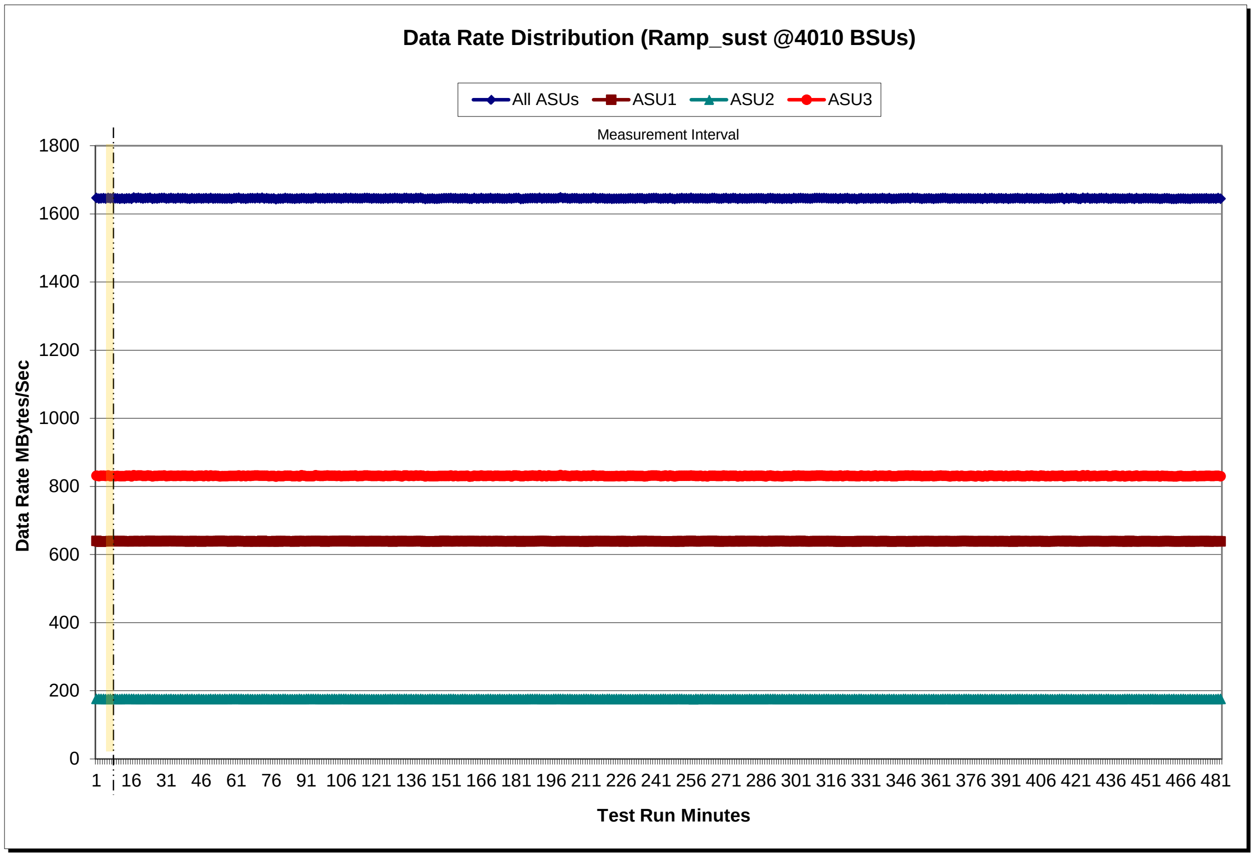
Sustainability Test Results File

Sustainability – Data Rate Distribution Data (MB/second)

The Sustainability Data Rate table of data is not embedded in this document due to its size. The table is available via the following URL:

Sustainability Data Rate Table

Sustainability – Data Rate Distribution Graph

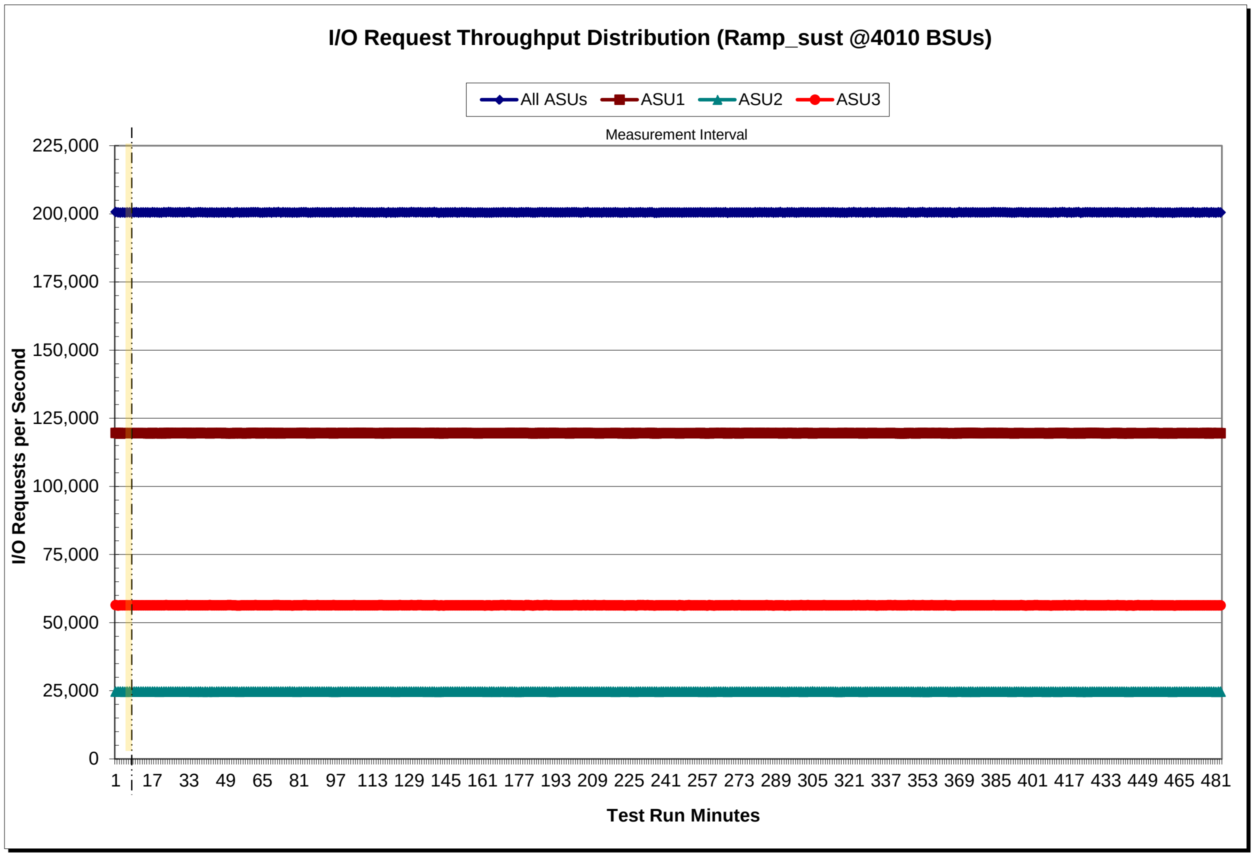


Sustainability – I/O Request Throughput Distribution Data

The Sustainability I/O Request Throughput table of data is not embedded in this document due to its size. The table is available via the following URL:

Sustainability I/O Request Throughput Table

Sustainability – I/O Request Throughput Distribution Graph

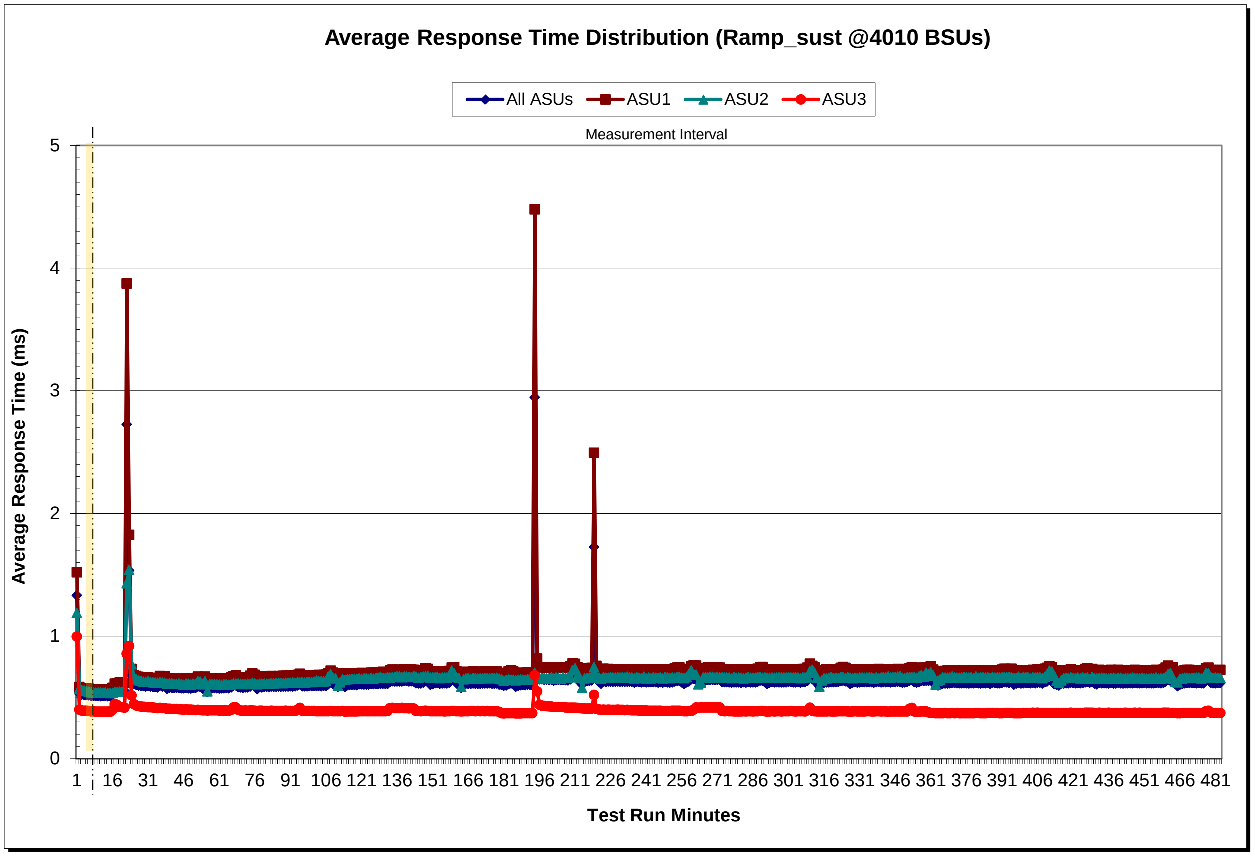


Sustainability – Average Response Time (ms) Distribution Data

The Sustainability Average Response Time table of data is not embedded in this document due to its size. The table is available via the following URL:

Sustainability Average Response Time Table

Sustainability – Average Response Time (ms) Distribution Graph



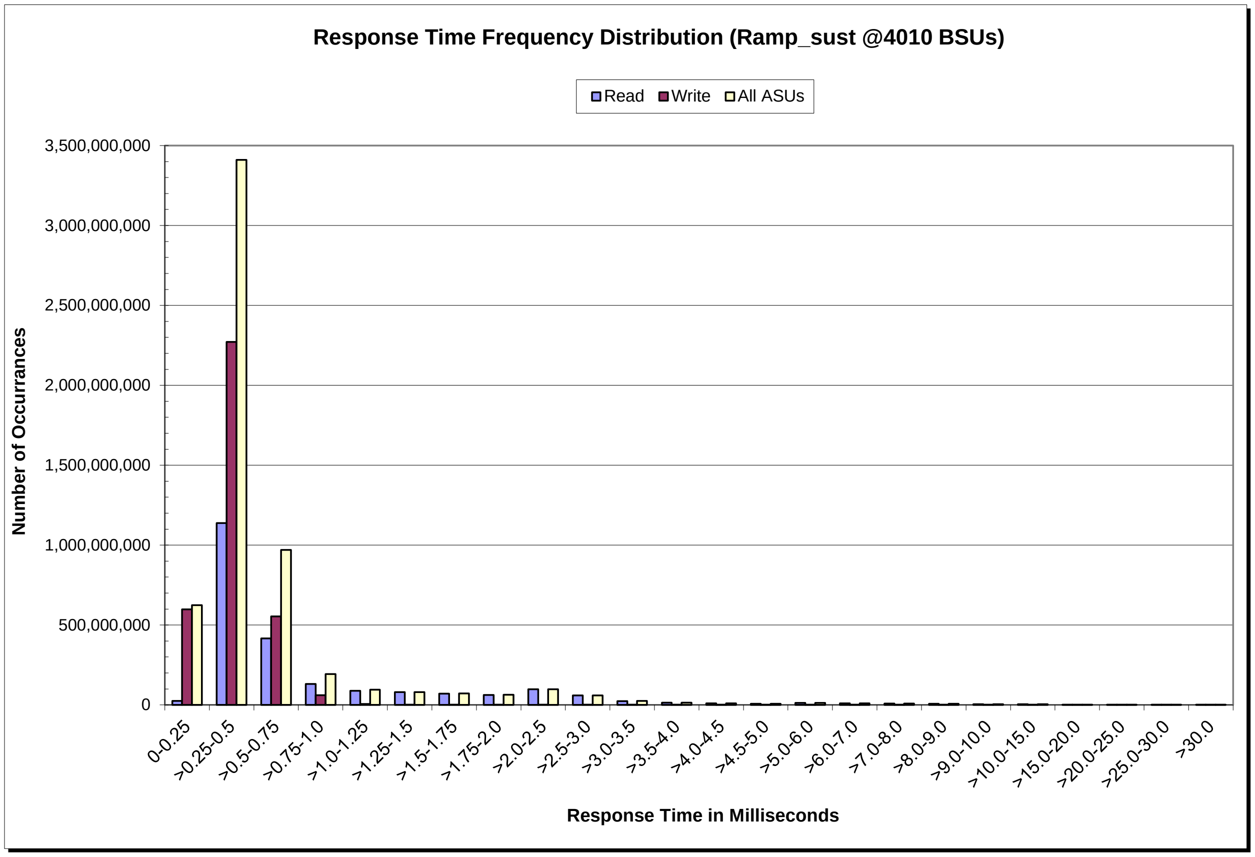
Sustainability – Response Time Frequency Distribution Data

Response Time (ms)	0-0.25	>0.25-0.5	>0.5-0.75	>0.75-1.0	>1.0-1.25	>1.25-1.5	>1.5-1.75	>1.75-2.0
Read	25,020,348	1,138,516,040	416,426,526	131,785,246	89,076,982	79,685,280	71,105,351	63,036,509
Write	598,668,784	2,272,189,571	554,246,380	61,085,971	6,174,735	1,217,422	565,101	386,311
All ASUs	623,689,132	3,410,705,611	970,672,906	192,871,217	95,251,717	80,902,702	71,670,452	63,422,820
ASU1	298,734,409	1,930,996,282	585,170,621	137,442,738	79,510,528	69,462,757	61,630,601	54,518,182
ASU2	80,238,112	415,720,964	108,035,728	24,604,247	12,607,286	10,773,426	9,687,471	8,649,910
ASU3	244,716,611	1,063,988,365	277,466,557	30,824,232	3,133,903	666,519	352,380	254,728

Response Time (ms)	>2.0-2.5	>2.5-3.0	>3.0-3.5	>3.5-4.0	>4.0-4.5	>4.5-5.0	>5.0-6.0	>6.0-7.0
Read	97,734,262	59,904,276	24,444,203	13,894,991	9,489,153	7,806,905	12,691,859	10,234,193
Write	478,429	267,335	165,546	124,021	100,353	103,863	305,336	417,034
All ASUs	98,212,691	60,171,611	24,609,749	14,019,012	9,589,506	7,910,768	12,997,195	10,651,227
ASU1	84,154,965	50,658,995	20,485,790	11,752,200	8,097,706	6,681,026	10,852,862	8,753,881
ASU2	13,739,969	9,342,472	4,024,551	2,194,894	1,434,497	1,172,339	1,993,471	1,702,215
ASU3	317,757	170,144	99,408	71,918	57,303	57,403	150,862	195,131

Response Time (ms)	>7.0-8.0	>8.0-9.0	>9.0-10.0	>10.0-15.0	>15.0-20.0	>20.0-25.0	>25.0-30.0	>30.0
Read	9,019,581	7,679,412	4,966,663	4,609,504	37,965	15,859	8,686	208,069
Write	213,332	47,653	10,379	25,304	24,614	16,022	9,403	186,415
All ASUs	9,232,913	7,727,065	4,977,042	4,634,808	62,579	31,881	18,089	394,484
ASU1	7,656,853	6,453,780	4,157,961	3,867,305	59,945	31,517	17,990	394,216
ASU2	1,479,329	1,252,327	814,983	763,345	2,351	122	94	218
ASU3	96,731	20,958	4,098	4,158	283	242	5	50

Sustainability – Response Time Frequency Distribution Graph



Sustainability – Measured Intensity Multiplier and Coefficient of Variation

<u>*Clause 3.4.3*</u>

IM – Intensity Multiplier: *The ratio of I/Os for each I/O stream relative to the total I/Os for all I/O streams (ASU1-1 – ASU3-1) as required by the benchmark specification.*

<u>*Clauses 5.1.10 and 5.3.15.2*</u>

MIM – Measured Intensity Multiplier: *The Measured Intensity Multiplier represents the ratio of measured I/Os for each I/O stream relative to the total I/Os measured for all I/O streams (ASU1-1 – ASU3-1). This value may differ from the corresponding Expected Intensity Multiplier by no more than 5%.*

<u>*Clause 5.3.15.3*</u>

COV – Coefficient of Variation: *This measure of variation for the Measured Intensity Multiplier cannot exceed 0.2.*

	ASU1-1	**ASU1-2**	**ASU1-3**	**ASU1-4**	**ASU2-1**	**ASU2-2**	**ASU2-3**	**ASU3-1**
IM	***0.0350***	***0.2810***	***0.0700***	***0.2100***	***0.0180***	***0.0700***	***0.0350***	***0.2810***
MIM	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
COV	0.002	0.000	0.001	0.001	0.002	0.001	0.001	0.000

Primary Metrics Test – IOPS Test Phase

Clause 5.4.4.2

The IOPS Test Phase consists of one Test Run at the 100% load point with a Measurement Interval of ten (10) minutes. The IOPS Test Phase immediately follows the Sustainability Test Phase without any interruption or manual intervention.

The IOPS Test Run generates the SPC-1 IOPS™ primary metric, which is computed as the I/O Request Throughput for the Measurement Interval of the IOPS Test Run.

The Average Response Time is computed for the IOPS Test Run and cannot exceed 30 milliseconds. If the Average Response Time exceeds the 30 millisecond constraint, the measurement is invalid.

Clause 9.4.3.7.3

For the IOPS Test Phase the FDR shall contain:

1. *I/O Request Throughput Distribution (data and graph).*
2. *A Response Time Frequency Distribution.*
3. *An Average Response Time Distribution.*
4. *The human readable Test Run Results File produced by the Workload Generator.*
5. *A listing or screen image of all input parameters supplied to the Workload Generator.*
6. *The total number of I/O Requests completed in the Measurement Interval as well as the number of I/O Requests with a Response Time less than or equal to 30 milliseconds and the number of I/O Requests with a Response Time greater than 30 milliseconds.*

SPC-1 Workload Generator Input Parameters

The SPC-1 Workload Generator input parameters for the Sustainability, IOPS, Response Time Ramp, Repeatability, and Persistence Test Runs are documented in Appendix E: SPC-1 Workload Generator Input Parameters on Page 76.

IOPS Test Results File

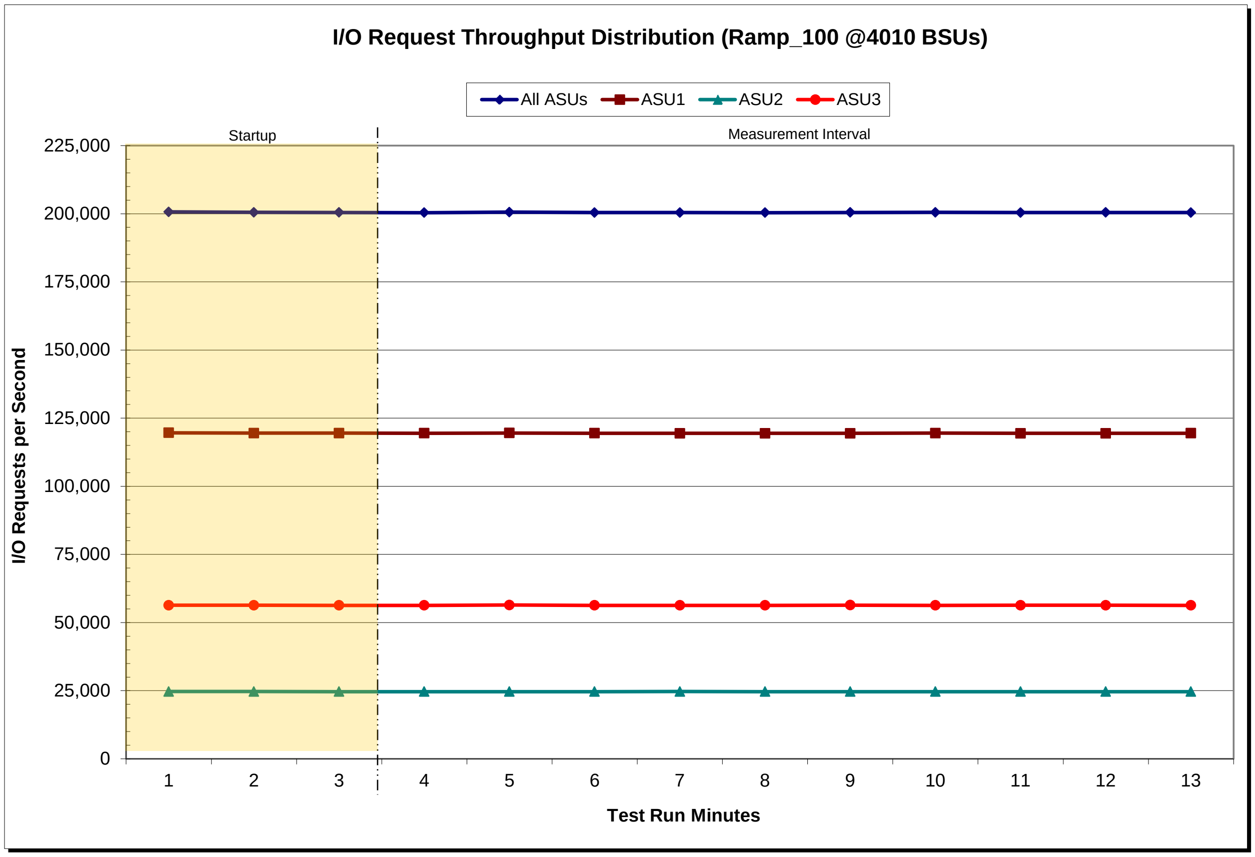
A link to the test results file generated from the IOPS Test Run is listed below.

IOPS Test Results File

IOPS Test Run – I/O Request Throughput Distribution Data

4,010 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	15:13:58	15:16:58	0-2	0:03:00
Measurement Interval	15:16:58	15:26:58	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	200,668.77	119,618.95	24,684.95	56,364.87
1	200,524.32	119,505.78	24,675.98	56,342.55
2	200,482.87	119,513.63	24,651.00	56,318.23
3	200,388.52	119,450.37	24,639.13	56,299.02
4	200,603.73	119,531.45	24,652.10	56,420.18
5	200,416.33	119,452.32	24,637.93	56,326.08
6	200,423.70	119,419.40	24,697.52	56,306.78
7	200,382.33	119,439.98	24,645.97	56,296.38
8	200,465.10	119,426.58	24,660.20	56,378.32
9	200,496.07	119,525.28	24,642.03	56,328.75
10	200,447.48	119,438.20	24,645.37	56,363.92
11	200,471.72	119,437.63	24,666.93	56,367.15
12	200,438.68	119,464.30	24,656.98	56,317.40
Average	***200,453.37***	***119,458.55***	***24,654.42***	***56,340.40***

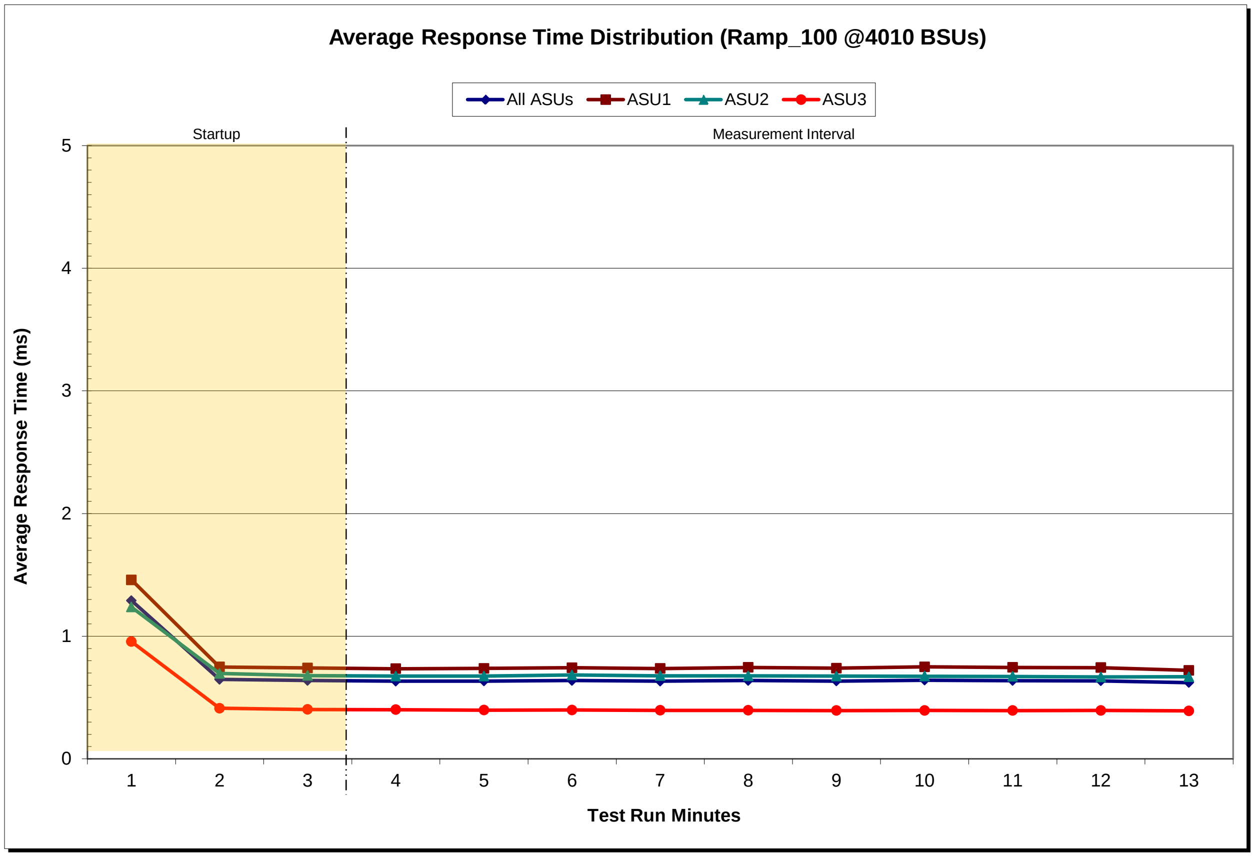
IOPS Test Run – I/O Request Throughput Distribution Graph



IOPS Test Run – Average Response Time (ms) Distribution Data

4,010 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	15:13:58	15:16:58	0-2	0:03:00
Measurement Interval	15:16:58	15:26:58	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	1.29	1.46	1.24	0.96
1	0.65	0.75	0.70	0.41
2	0.64	0.74	0.68	0.40
3	0.63	0.73	0.67	0.40
4	0.63	0.74	0.67	0.40
5	0.64	0.74	0.68	0.40
6	0.63	0.74	0.68	0.40
7	0.64	0.75	0.68	0.39
8	0.63	0.74	0.67	0.39
9	0.64	0.75	0.67	0.39
10	0.64	0.75	0.67	0.39
11	0.64	0.74	0.67	0.39
12	0.62	0.72	0.67	0.39
Average	***0.63***	***0.74***	***0.67***	***0.40***

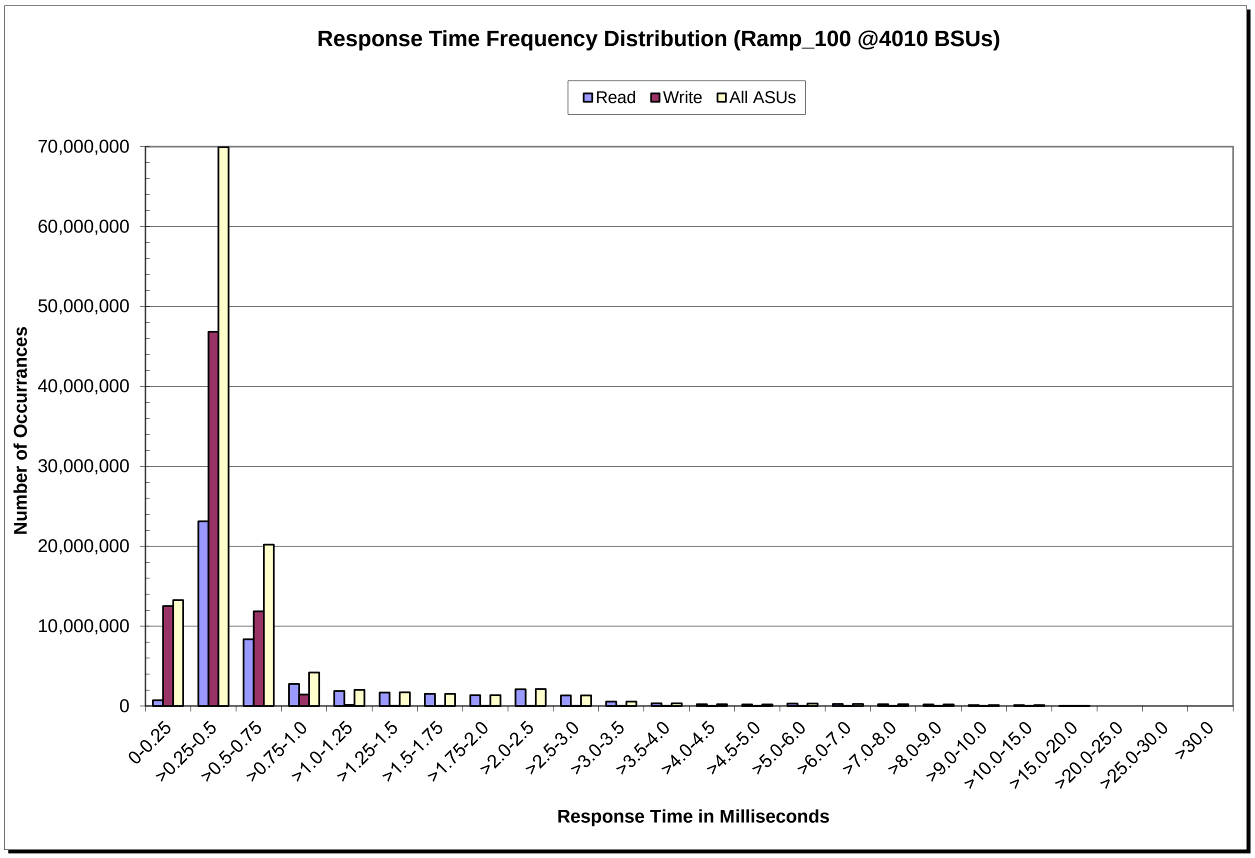
IOPS Test Run – Average Response Time (ms) Distribution Graph



IOPS Test Run –Response Time Frequency Distribution Data

Response Time (ms)	0-0.25	>0.25-0.5	>0.5-0.75	>0.75-1.0	>1.0-1.25	>1.25-1.5	>1.5-1.75	>1.75-2.0
Read	735,387	23,118,642	8,362,658	2,756,473	1,883,109	1,692,301	1,514,481	1,349,108
Write	12,516,565	46,825,037	11,840,411	1,435,268	148,933	26,184	11,733	8,743
All ASUs	13,251,952	69,943,679	20,203,069	4,191,741	2,032,042	1,718,485	1,526,214	1,357,851
ASU1	6,469,755	39,330,320	12,153,537	2,972,388	1,691,852	1,473,916	1,310,902	1,165,594
ASU2	1,845,505	8,635,523	2,014,053	487,374	264,265	230,093	207,793	186,467
ASU3	4,936,692	21,977,836	6,035,479	731,979	75,925	14,476	7,519	5,790
Response Time (ms)	**>2.0-2.5**	**>2.5-3.0**	**>3.0-3.5**	**>3.5-4.0**	**>4.0-4.5**	**>4.5-5.0**	**>5.0-6.0**	**>6.0-7.0**
Read	2,109,121	1,323,758	566,303	334,506	234,060	195,463	318,971	259,645
Write	11,544	6,863	4,100	2,980	2,278	1,688	2,572	1,939
All ASUs	2,120,665	1,330,621	570,403	337,486	236,338	197,151	321,543	261,584
ASU1	1,813,413	1,119,200	475,326	283,383	199,613	166,605	270,484	218,927
ASU2	299,786	207,333	92,913	52,643	35,799	29,902	50,252	42,241
ASU3	7,466	4,088	2,164	1,460	926	644	807	416
Response Time (ms)	**>7.0-8.0**	**>8.0-9.0**	**>9.0-10.0**	**>10.0-15.0**	**>15.0-20.0**	**>20.0-25.0**	**>25.0-30.0**	**>30.0**
Read	228,787	195,225	126,836	117,542	383	4	-	-
Write	1,581	392	137	157	12	2	1	-
All ASUs	230,368	195,617	126,973	117,699	395	6	1	-
ASU1	192,347	163,521	105,853	97,760	329	6	1	-
ASU2	37,658	31,999	21,102	19,875	60	-	-	-
ASU3	363	97	18	64	6	-	-	-

IOPS Test Run –Response Time Frequency Distribution Graph



IOPS Test Run – I/O Request Information

I/O Requests Completed in the Measurement Interval	I/O Requests Completed with Response Time = or < 30 ms	I/O Requests Completed with Response Time > 30 ms
120,271,883	120,271,883	0

IOPS Test Run – Measured Intensity Multiplier and Coefficient of Variation

<u>*Clause 3.4.3*</u>

IM – Intensity Multiplier: *The ratio of I/Os for each I/O stream relative to the total I/Os for all I/O streams (ASU1-1 – ASU3-1) as required by the benchmark specification.*

<u>*Clauses 5.1.10 and 5.3.15.2*</u>

MIM – Measured Intensity Multiplier: *The Measured Intensity Multiplier represents the ratio of measured I/Os for each I/O stream relative to the total I/Os measured for all I/O streams (ASU1-1 – ASU3-1). This value may differ from the corresponding Expected Intensity Multiplier by no more than 5%.*

<u>*Clause 5.3.15.3*</u>

COV – Coefficient of Variation: *This measure of variation for the Measured Intensity Multiplier cannot exceed 0.2.*

	ASU1-1	ASU1-2	ASU1-3	ASU1-4	ASU2-1	ASU2-2	ASU2-3	ASU3-1
IM	***0.0350***	***0.2810***	***0.0700***	***0.2100***	***0.0180***	***0.0700***	***0.0350***	***0.2810***
MIM	0.0350	0.2809	0.0700	0.2101	0.0180	0.0700	0.0350	0.2811
COV	0.001	0.000	0.001	0.001	0.003	0.001	0.002	0.000

Primary Metrics Test – Response Time Ramp Test Phase

Clause 5.4.4.3

The Response Time Ramp Test Phase consists of five Test Runs, one each at 95%, 90%, 80%, 50%, and 10% of the load point (100%) used to generate the SPC-1 IOPS™ primary metric. Each of the five Test Runs has a Measurement Interval of ten (10) minutes. The Response Time Ramp Test Phase immediately follows the IOPS Test Phase without any interruption or manual intervention.

The five Response Time Ramp Test Runs, in conjunction with the IOPS Test Run (100%), demonstrate the relationship between Average Response Time and I/O Request Throughput for the Tested Storage Configuration (TSC) as illustrated in the response time/throughput curve on page 17.

In addition, the Average Response Time measured during the 10% Test Run is the value for the SPC-1 LRT™ metric. That value represents the Average Response Time of a lightly loaded TSC.

Clause 9.4.3.7.4

The following content shall appear in the FDR for the Response Time Ramp Phase:

1. *A Response Time Ramp Distribution.*
2. *The human readable Test Run Results File produced by the Workload Generator for each Test Run within the Response Time Ramp Test Phase.*
3. *For the 10% Load Level Test Run (SPC-1 LRT™ metric) an Average Response Time Distribution.*
4. *A listing or screen image of all input parameters supplied to the Workload Generator.*

SPC-1 Workload Generator Input Parameters

The SPC-1 Workload Generator input parameters for the Sustainability, IOPS, Response Time Ramp, Repeatability, and Persistence Test Runs are documented in Appendix E: SPC-1 Workload Generator Input Parameters on Page 76.

Response Time Ramp Test Results File

A link to each test result file generated from each Response Time Ramp Test Run list listed below.

95% Load Level

90% Load Level

80% Load Level

50% Load Level

10% Load Level

Response Time Ramp Distribution (IOPS) Data

The five Test Runs that comprise the Response Time Ramp Phase are executed at 95%, 90%, 80%, 50%, and 10% of the Business Scaling Unit (BSU) load level used to produce the SPC-1 IOPS™ primary metric. The 100% BSU load level is included in the following Response Time Ramp data table and graph for completeness.

100% Load Level:					**95% Load Level:**				
4,010 BSUs	**Start**	**Stop**	**Interval**	**Duration**	**3,809 BSUs**	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	15:13:58	15:16:58	0-3	0:03:00	**Start-Up/Ramp-Up**	15:28:25	15:31:25	0-3	0:03:00
Measurement Interval	15:16:58	15:26:58	3-12	0:10:00	**Measurement Interval**	15:31:25	15:41:25	3-12	0:10:00
(60 second intervals)	**All ASUs**	**ASU-1**	**ASU-2**	**ASU-3**	*(60 second intervals)*	**All ASUs**	**ASU-1**	**ASU-2**	**ASU-3**
0	200,668.77	119,618.95	24,684.95	56,364.87	**0**	190,633.13	113,634.62	23,451.43	53,547.08
1	200,524.32	119,505.78	24,675.98	56,342.55	**1**	190,468.78	113,501.57	23,434.48	53,532.73
2	200,482.87	119,513.63	24,651.00	56,318.23	**2**	190,413.07	113,473.23	23,402.70	53,537.13
3	200,388.52	119,450.37	24,639.13	56,299.02	**3**	190,478.02	113,528.90	23,414.87	53,534.25
4	200,603.73	119,531.45	24,652.10	56,420.18	**4**	190,475.18	113,439.63	23,469.48	53,566.07
5	200,416.33	119,452.32	24,637.93	56,326.08	**5**	190,479.75	113,528.38	23,434.65	53,516.72
6	200,423.70	119,419.40	24,697.52	56,306.78	**6**	190,607.63	113,607.55	23,436.55	53,563.53
7	200,382.33	119,439.98	24,645.97	56,296.38	**7**	190,494.60	113,581.45	23,446.87	53,466.28
8	200,465.10	119,426.58	24,660.20	56,378.32	**8**	190,536.33	113,496.63	23,448.68	53,591.02
9	200,496.07	119,525.28	24,642.03	56,328.75	**9**	190,446.55	113,505.67	23,445.42	53,495.47
10	200,447.48	119,438.20	24,645.37	56,363.92	**10**	190,478.03	113,539.62	23,392.08	53,546.33
11	200,471.72	119,437.63	24,666.93	56,367.15	**11**	190,437.57	113,536.32	23,430.53	53,470.72
12	200,438.68	119,464.30	24,656.98	56,317.40	**12**	190,485.75	113,530.15	23,439.75	53,515.85
Average	***200,453.37***	***119,458.55***	***24,654.42***	***56,340.40***	***Average***	***190,491.94***	***113,529.43***	***23,435.89***	***53,526.62***

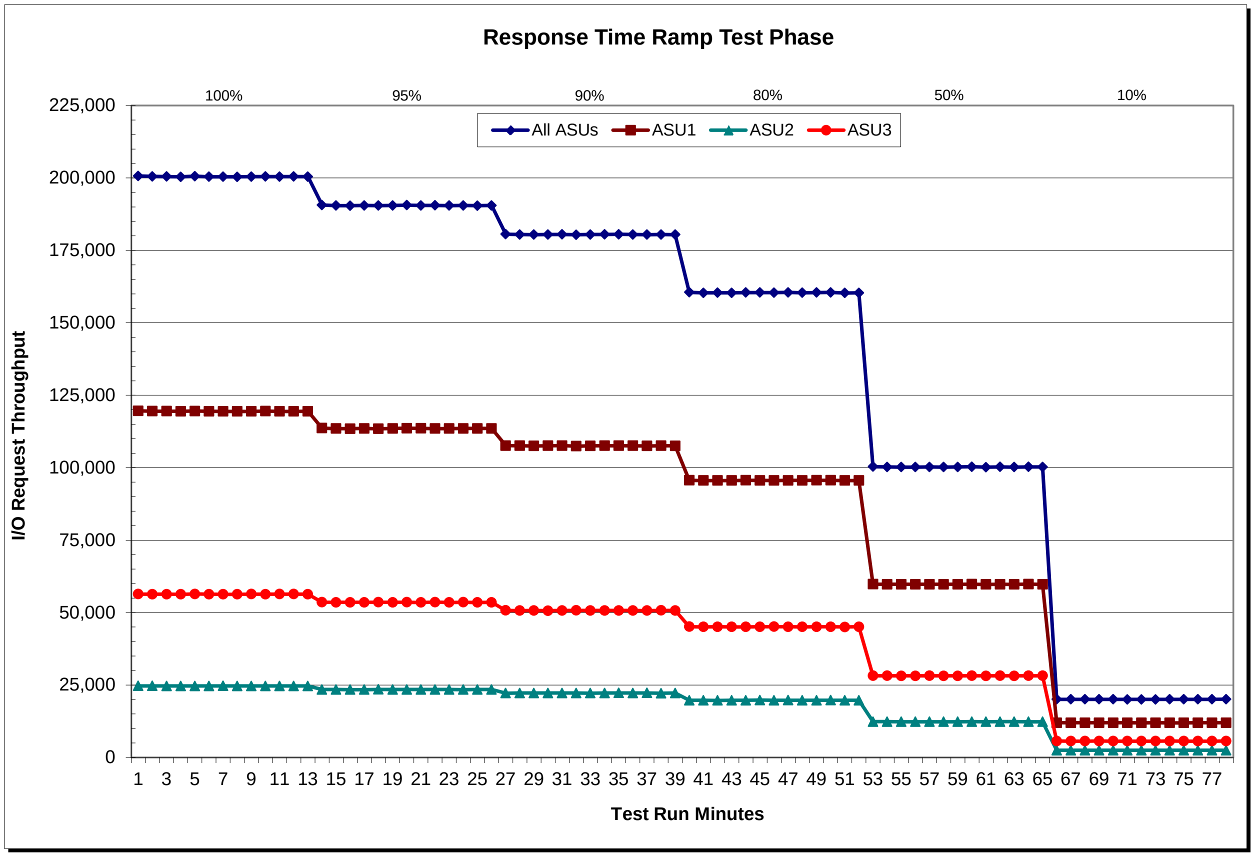
90% Load Level:					**80% Load Level:**				
3,609 BSUs	**Start**	**Stop**	**Interval**	**Duration**	**3,208 BSUs**	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	15:42:52	15:45:52	0-3	0:03:00	**Start-Up/Ramp-Up**	15:57:17	16:00:17	0-3	0:03:00
Measurement Interval	15:45:52	15:55:52	3-12	0:10:00	**Measurement Interval**	16:00:17	16:10:17	3-12	0:10:00
(60 second intervals)	**All ASUs**	**ASU-1**	**ASU-2**	**ASU-3**	*(60 second intervals)*	**All ASUs**	**ASU-1**	**ASU-2**	**ASU-3**
0	180,570.42	107,620.93	22,202.47	50,747.02	**0**	160,535.47	95,661.22	19,732.38	45,141.87
1	180,476.72	107,586.75	22,205.85	50,684.12	**1**	160,349.08	95,585.73	19,724.45	45,038.90
2	180,434.65	107,507.68	22,220.15	50,706.82	**2**	160,357.82	95,561.22	19,693.88	45,102.72
3	180,448.52	107,589.78	22,209.98	50,648.75	**3**	160,338.12	95,560.02	19,724.62	45,053.48
4	180,515.05	107,609.90	22,211.62	50,693.53	**4**	160,437.05	95,635.73	19,733.12	45,068.20
5	180,396.80	107,438.62	22,227.93	50,730.25	**5**	160,467.08	95,605.28	19,778.67	45,083.13
6	180,472.92	107,552.75	22,201.68	50,718.48	**6**	160,405.73	95,569.57	19,706.98	45,129.18
7	180,513.63	107,582.60	22,217.82	50,713.22	**7**	160,474.12	95,616.92	19,756.67	45,100.53
8	180,535.53	107,567.68	22,246.63	50,721.22	**8**	160,382.60	95,594.12	19,730.25	45,058.23
9	180,466.12	107,557.13	22,214.05	50,694.93	**9**	160,454.65	95,628.12	19,729.25	45,097.28
10	180,411.55	107,501.90	22,245.03	50,664.62	**10**	160,494.82	95,653.75	19,758.52	45,082.55
11	180,480.35	107,574.28	22,163.38	50,742.68	**11**	160,314.50	95,574.17	19,710.57	45,029.77
12	180,438.85	107,535.80	22,203.73	50,699.32	**12**	160,353.47	95,569.85	19,698.78	45,084.83
Average	***180,467.93***	***107,551.05***	***22,214.19***	***50,702.70***	***Average***	***160,412.21***	***95,600.75***	***19,732.74***	***45,078.72***

Response Time Ramp Distribution (IOPS) Data *(continued)*

50% Load Level: 2,005 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	16:11:39	16:14:39	0-3	0:03:00
Measurement Interval	16:14:39	16:24:39	3-12	0:10:00
(60 second intervals)	**All ASUs**	**ASU-1**	**ASU-2**	**ASU-3**
0	100,341.67	59,804.70	12,358.78	28,178.18
1	100,236.93	59,736.93	12,333.15	28,166.85
2	100,223.13	59,758.58	12,322.53	28,142.02
3	100,216.95	59,761.22	12,313.13	28,142.60
4	100,246.93	59,753.95	12,322.40	28,170.58
5	100,215.78	59,738.78	12,319.53	28,157.47
6	100,231.33	59,763.03	12,316.97	28,151.33
7	100,303.05	59,774.77	12,338.40	28,189.88
8	100,173.03	59,718.63	12,323.85	28,130.55
9	100,289.65	59,763.27	12,332.62	28,193.77
10	100,206.55	59,726.57	12,336.58	28,143.40
11	100,288.87	59,809.12	12,310.38	28,169.37
12	100,235.87	59,709.80	12,324.93	28,201.13
Average	***100,240.80***	***59,751.91***	***12,323.88***	***28,165.01***

10% Load Level: 401 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	16:25:56	16:28:56	0-3	0:03:00
Measurement Interval	16:28:56	16:38:56	3-12	0:10:00
(60 second intervals)	**All ASUs**	**ASU-1**	**ASU-2**	**ASU-3**
0	20,066.83	11,952.95	2,458.23	5,655.65
1	20,052.43	11,954.28	2,471.32	5,626.83
2	20,056.17	11,970.45	2,460.98	5,624.73
3	20,071.50	11,965.68	2,462.75	5,643.07
4	20,060.38	11,952.25	2,470.48	5,637.65
5	20,037.32	11,951.47	2,459.17	5,626.68
6	20,034.75	11,944.40	2,459.95	5,630.40
7	20,031.77	11,957.32	2,451.63	5,622.82
8	20,045.23	11,936.82	2,464.22	5,644.20
9	20,038.70	11,942.58	2,457.02	5,639.10
10	20,053.12	11,964.60	2,466.25	5,622.27
11	20,063.03	11,963.07	2,465.75	5,634.22
12	20,097.57	11,967.48	2,484.75	5,645.33
Average	***20,053.34***	***11,954.57***	***2,464.20***	***5,634.57***

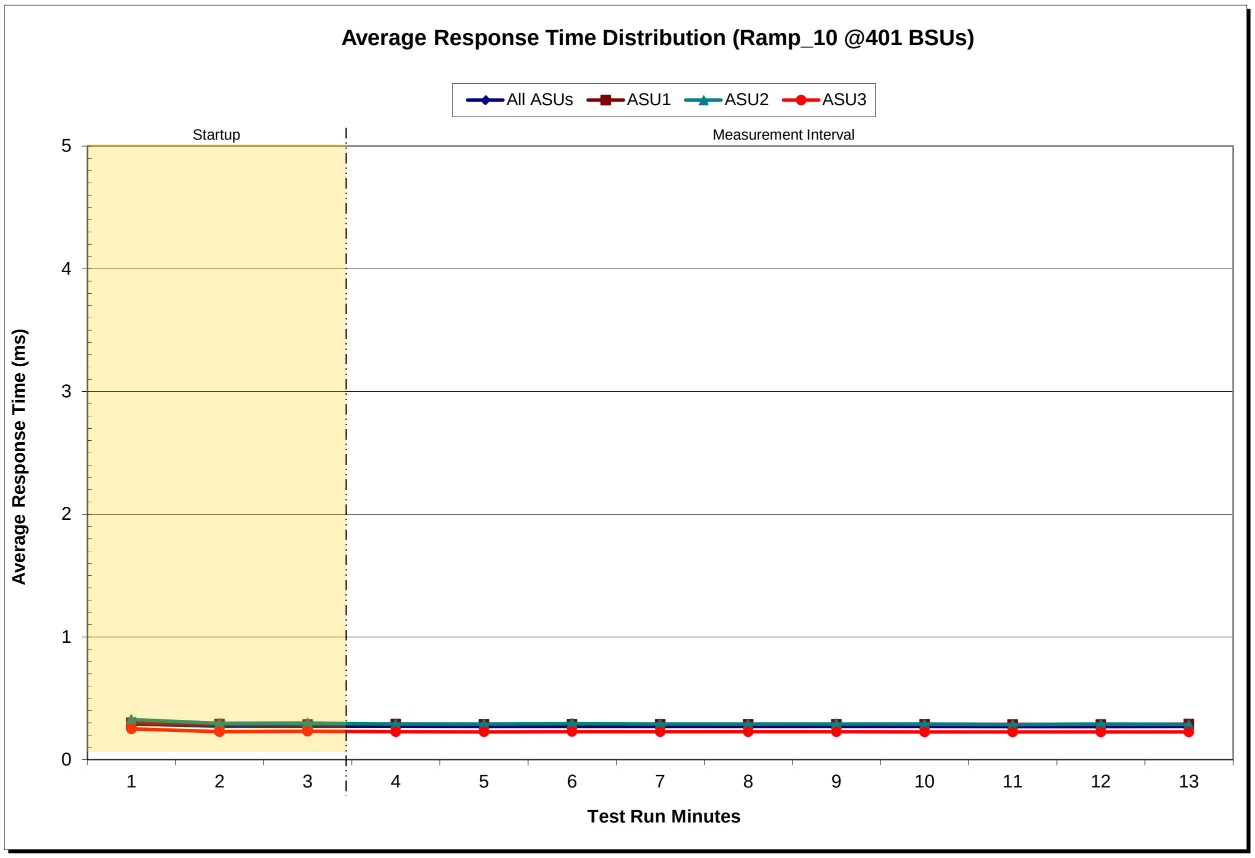
Response Time Ramp Distribution (IOPS) Graph



SPC-1 LRT™ Average Response Time (ms) Distribution Data

401 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	16:25:56	16:28:56	0-2	0:03:00
Measurement Interval	16:28:56	16:38:56	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	0.29	0.30	0.33	0.25
1	0.27	0.29	0.30	0.23
2	0.27	0.29	0.30	0.23
3	0.27	0.29	0.29	0.23
4	0.27	0.29	0.29	0.23
5	0.27	0.29	0.29	0.23
6	0.27	0.29	0.29	0.23
7	0.27	0.29	0.29	0.23
8	0.27	0.29	0.29	0.23
9	0.27	0.29	0.29	0.23
10	0.27	0.29	0.29	0.23
11	0.27	0.29	0.29	0.23
12	0.27	0.29	0.29	0.23
Average	***0.27***	***0.29***	***0.29***	***0.23***

SPC-1 LRT™ Average Response Time (ms) Distribution Graph



SPC-1 LRT™ (10%) – Measured Intensity Multiplier and Coefficient of Variation

<u>*Clause 3.4.3*</u>

IM – Intensity Multiplier: *The ratio of I/Os for each I/O stream relative to the total I/Os for all I/O streams (ASU1-1 – ASU3-1) as required by the benchmark specification.*

<u>*Clauses 5.1.10 and 5.3.15.2*</u>

MIM – Measured Intensity Multiplier: *The Measured Intensity Multiplier represents the ratio of measured I/Os for each I/O stream relative to the total I/Os measured for all I/O streams (ASU1-1 – ASU3-1). This value may differ from the corresponding Expected Intensity Multiplier by no more than 5%.*

<u>*Clause 5.3.15.3*</u>

COV – Coefficient of Variation: *This measure of variation for the Measured Intensity Multiplier cannot exceed 0.2.*

	ASU1-1	**ASU1-2**	**ASU1-3**	**ASU1-4**	**ASU2-1**	**ASU2-2**	**ASU2-3**	**ASU3-1**
IM	***0.0350***	***0.2810***	***0.0700***	***0.2100***	***0.0180***	***0.0700***	***0.0350***	***0.2810***
MIM	0.0351	0.2809	0.0701	0.2101	0.0180	0.0699	0.0350	0.2810
COV	0.004	0.001	0.004	0.002	0.006	0.004	0.003	0.001

Repeatability Test

Clause 5.4.5

The Repeatability Test demonstrates the repeatability and reproducibility of the SPC-1 IOPS™ primary metric and the SPC-1 LRT™ metric generated in earlier Test Runs.

There are two identical Repeatability Test Phases. Each Test Phase contains two Test Runs. Each of the Test Runs will have a Measurement Interval of no less than ten (10) minutes. The two Test Runs in each Test Phase will be executed without interruption or any type of manual intervention.

The first Test Run in each Test Phase is executed at the 10% load point. The Average Response Time from each of the Test Runs is compared to the SPC-1 LRT™ metric. Each Average Response Time value must be less than the SPC-1 LRT™ metric plus 5% or less than the SPC-1 LRT™ metric plus one (1) millisecond (ms).

The second Test Run in each Test Phase is executed at the 100% load point. The I/O Request Throughput from the Test Runs is compared to the SPC-1 IOPS™ primary metric. Each I/O Request Throughput value must be greater than the SPC-1 IOPS™ primary metric minus 5%. In addition, the Average Response Time for each Test Run cannot exceed 30 milliseconds.

If any of the above constraints are not met, the benchmark measurement is invalid.

Clause 9.4.3.7.5

The following content shall appear in the FDR for each Test Run in the two Repeatability Test Phases:

1. *A table containing the results of the Repeatability Test.*
2. *An I/O Request Throughput Distribution graph and table.*
3. *An Average Response Time Distribution graph and table.*
4. *The human readable Test Run Results File produced by the Workload Generator.*
5. *A listing or screen image of all input parameters supplied to the Workload Generator.*

SPC-1 Workload Generator Input Parameters

The SPC-1 Workload Generator input parameters for the Sustainability, IOPS, Response Time Ramp, Repeatability, and Persistence Test Runs are documented in Appendix E: SPC-1 Workload Generator Input Parameters on Page 76.

Repeatability Test Results File

The values for the SPC-1 IOPS™, SPC-1 LRT™, and the Repeatability Test measurements are listed in the tables below.

	SPC-1 IOPS™
Primary Metrics	***200,453.37***
Repeatability Test Phase 1	200,501.38
Repeatability Test Phase 2	200,483.38

The SPC-1 IOPS™ values in the above table were generated using 100% of the specified Business Scaling Unit (BSU) load level. Each of the Repeatability Test Phase values for SPC-1 IOPS™ must greater than 95% of the reported SPC-1 IOPS™ Primary Metric.

	SPC-1 LRT™
Primary Metrics	***0.27 ms***
Repeatability Test Phase 1	0.27 ms
Repeatability Test Phase 2	0.27 ms

The average response time values in the SPC-1 LRT™ column were generated using 10% of the specified Business Scaling Unit (BSU) load level. Each of the Repeatability Test Phase values for SPC-1 LRT™ must be less than 105% of the reported SPC-1 LRT™ Primary Metric or less than the reported SPC-1 LRT™ Primary Metric plus one (1) millisecond (ms).

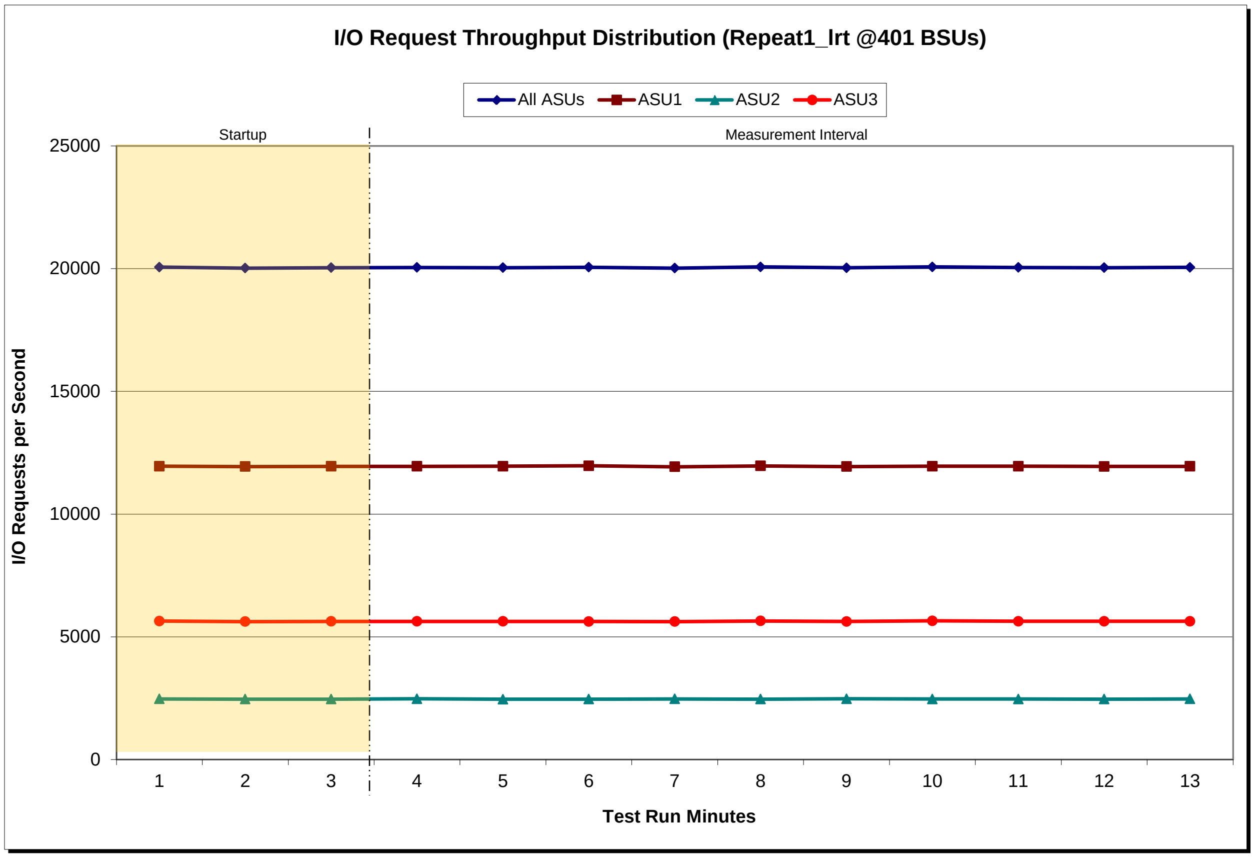
A link to the test result file generated from each Repeatability Test Run is listed below.

Repeatability Test Phase 1, Test Run 1 (LRT)
Repeatability Test Phase 1, Test Run 2 (IOPS)
Repeatability Test Phase 2, Test Run 1 (LRT)
Repeatability Test Phase 2, Test Run 2 (IOPS)

Repeatability 1 LRT – I/O Request Throughput Distribution Data

401 BSUs	Start	Stop	Interval	Duration
Start-Up/Ramp-Up	16:40:23	16:43:23	0-2	0:03:00
Measurement Interval	16:43:23	16:53:23	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	20,063.62	11,949.75	2,472.37	5,641.50
1	20,020.43	11,938.78	2,463.15	5,618.50
2	20,039.00	11,947.85	2,461.53	5,629.62
3	20,050.50	11,944.28	2,475.47	5,630.75
4	20,039.58	11,952.80	2,458.87	5,627.92
5	20,058.05	11,969.25	2,462.33	5,626.47
6	20,021.77	11,930.12	2,471.60	5,620.05
7	20,070.22	11,962.92	2,459.72	5,647.58
8	20,035.65	11,938.35	2,474.78	5,622.52
9	20,070.98	11,952.33	2,467.73	5,650.92
10	20,048.95	11,949.73	2,465.50	5,633.72
11	20,040.80	11,943.73	2,462.85	5,634.22
12	20,051.80	11,945.38	2,472.22	5,634.20
Average	***20,048.83***	***11,948.89***	***2,467.11***	***5,632.83***

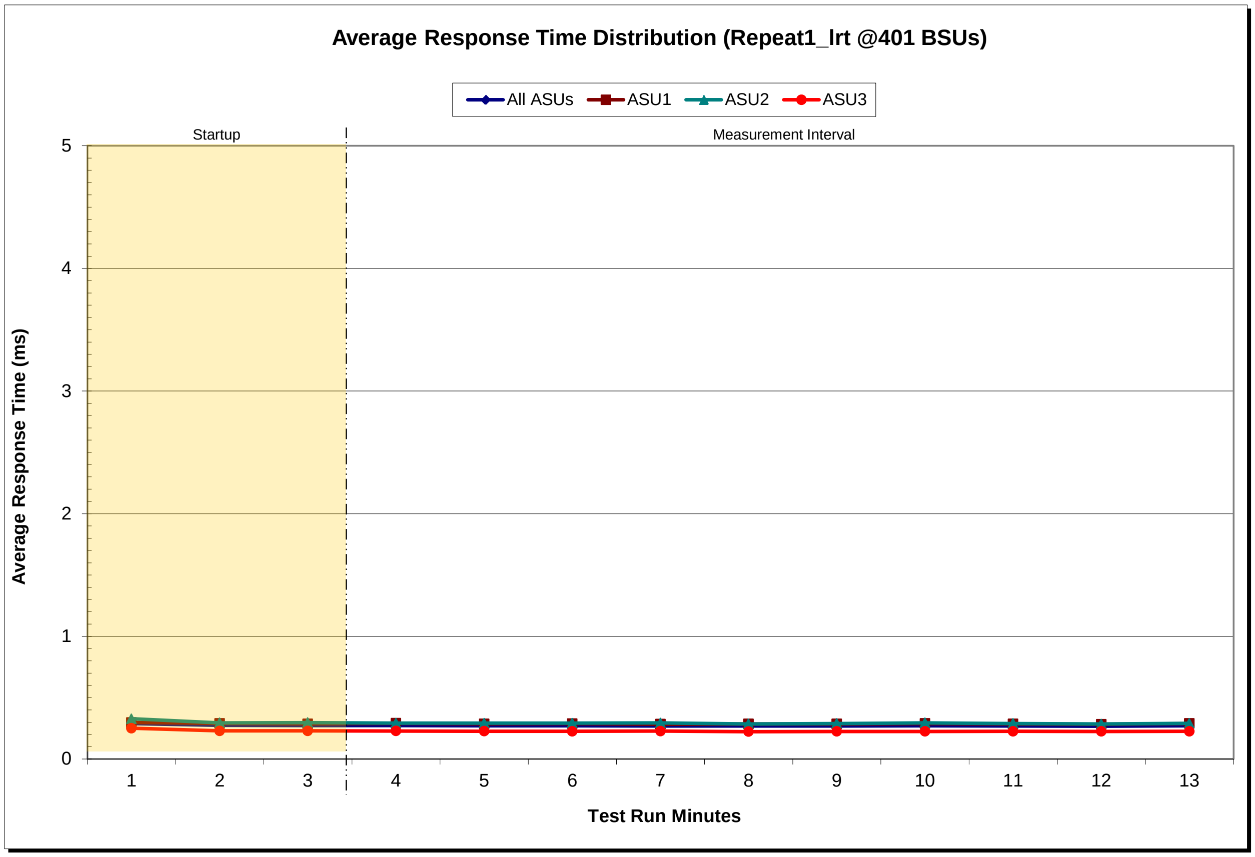
Repeatability 1 LRT – I/O Request Throughput Distribution Graph



Repeatability 1 LRT –Average Response Time (ms) Distribution Data

401 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	16:40:23	16:43:23	0-2	0:03:00
Measurement Interval	16:43:23	16:53:23	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	0.29	0.30	0.33	0.25
1	0.27	0.29	0.29	0.23
2	0.27	0.29	0.30	0.23
3	0.27	0.29	0.29	0.23
4	0.27	0.29	0.29	0.23
5	0.27	0.29	0.29	0.23
6	0.27	0.28	0.29	0.23
7	0.27	0.29	0.29	0.22
8	0.27	0.29	0.29	0.22
9	0.27	0.29	0.29	0.23
10	0.27	0.29	0.29	0.23
11	0.27	0.28	0.29	0.23
12	0.27	0.29	0.29	0.23
Average	***0.27***	***0.29***	***0.29***	***0.23***

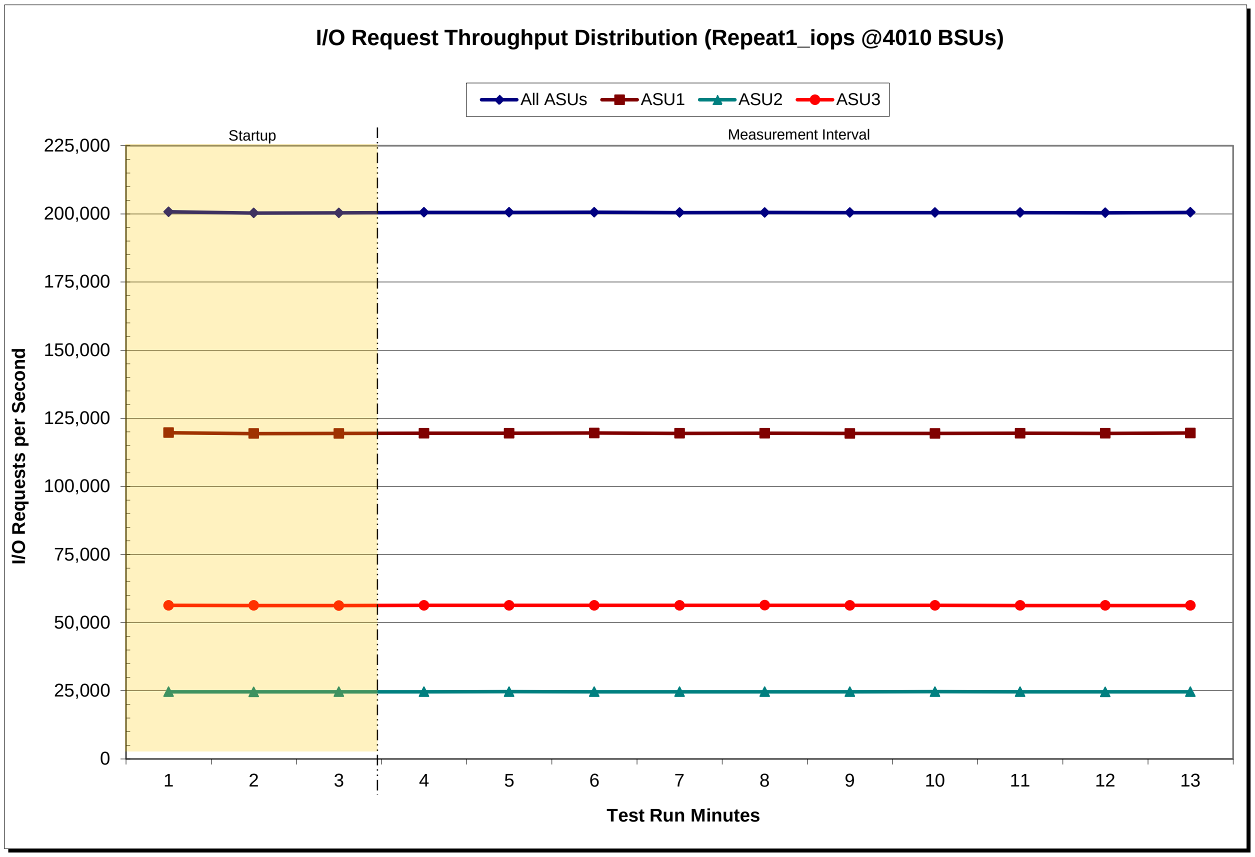
Repeatability 1 LRT –Average Response Time (ms) Distribution Graph



Repeatability 1 IOPS – I/O Request Throughput Distribution Data

4,010 BSUs	Start	Stop	Interval	Duration
Start-Up/Ramp-Up	16:54:50	16:57:50	0-2	0:03:00
Measurement Interval	16:57:50	17:07:50	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	200,746.12	119,717.63	24,670.93	56,357.55
1	200,327.95	119,399.70	24,628.18	56,300.07
2	200,338.62	119,414.87	24,656.82	56,266.93
3	200,534.80	119,489.10	24,673.30	56,372.40
4	200,538.28	119,518.87	24,684.27	56,335.15
5	200,593.82	119,593.53	24,640.63	56,359.65
6	200,468.23	119,456.53	24,642.55	56,369.15
7	200,519.70	119,491.87	24,648.80	56,379.03
8	200,479.67	119,440.37	24,674.05	56,365.25
9	200,469.55	119,416.92	24,681.02	56,371.62
10	200,476.83	119,493.70	24,671.38	56,311.75
11	200,384.37	119,467.00	24,612.67	56,304.70
12	200,548.53	119,587.63	24,655.50	56,305.40
Average	***200,501.38***	***119,495.55***	***24,658.42***	***56,347.41***

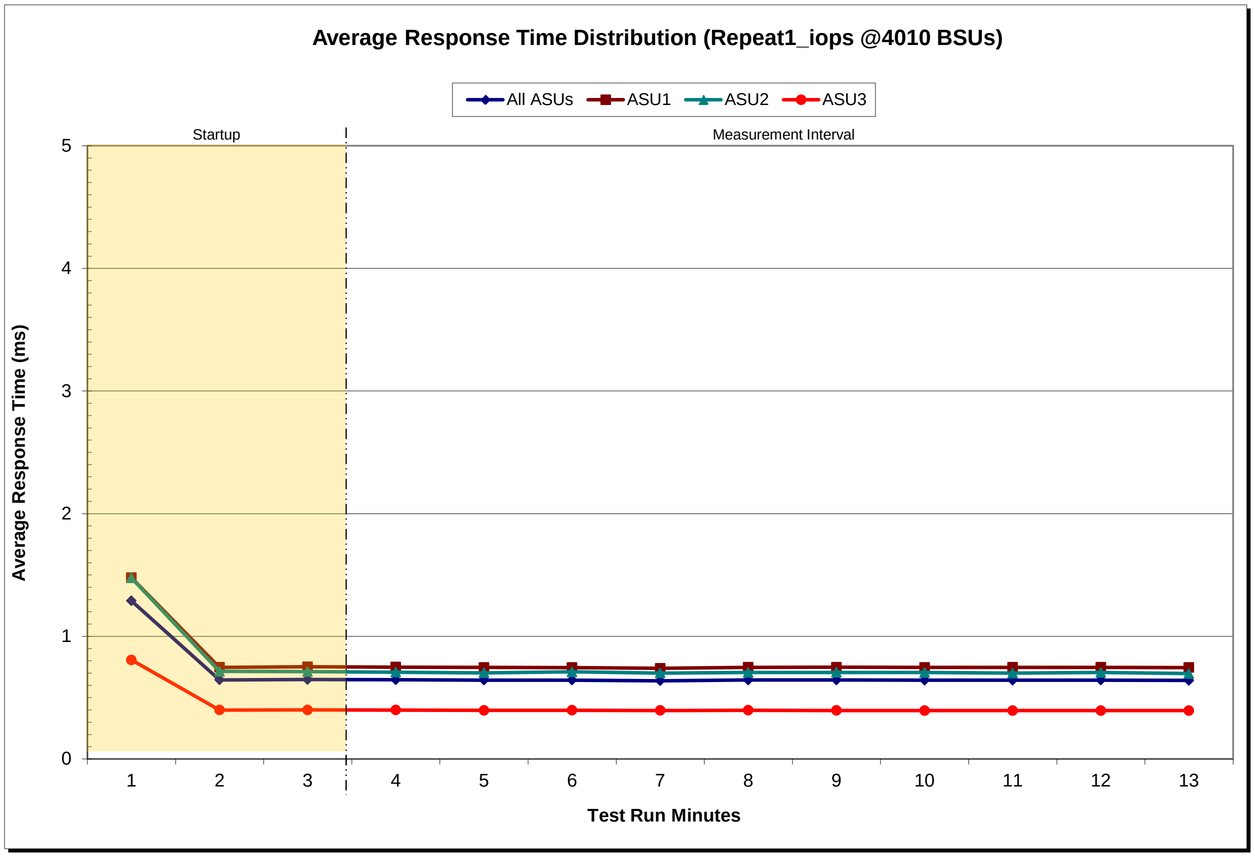
Repeatability 1 IOPS – I/O Request Throughput Distribution Graph



Repeatability 1 IOPS –Average Response Time (ms) Distribution Data

4,010 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	16:54:50	16:57:50	0-2	0:03:00
Measurement Interval	16:57:50	17:07:50	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	1.29	1.48	1.48	0.81
1	0.65	0.75	0.71	0.40
2	0.65	0.75	0.71	0.40
3	0.65	0.75	0.71	0.40
4	0.64	0.75	0.70	0.40
5	0.64	0.75	0.71	0.40
6	0.64	0.74	0.70	0.40
7	0.64	0.75	0.71	0.40
8	0.64	0.75	0.71	0.40
9	0.64	0.75	0.71	0.39
10	0.64	0.75	0.70	0.39
11	0.64	0.75	0.71	0.39
12	0.64	0.75	0.70	0.39
Average	***0.64***	***0.75***	***0.70***	***0.40***

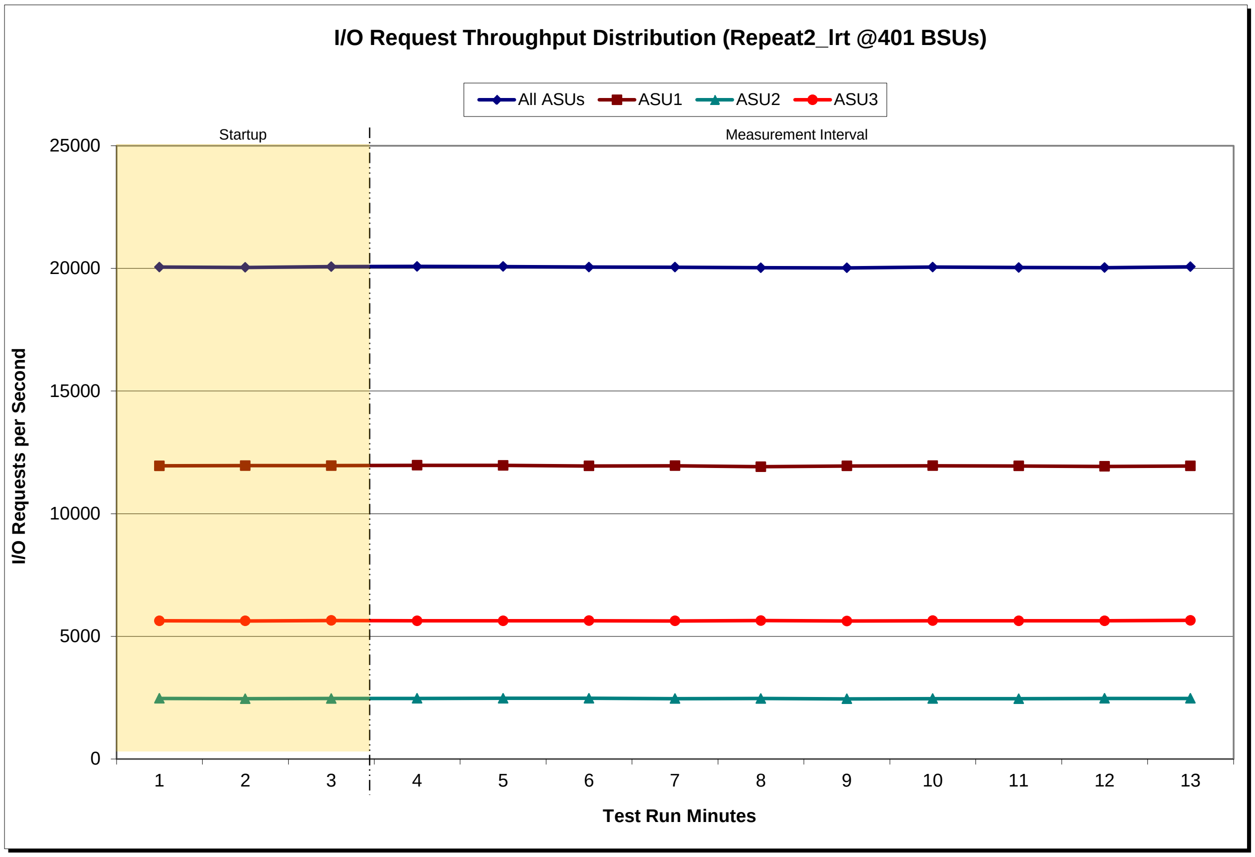
Repeatability 1 IOPS –Average Response Time (ms) Distribution Graph



Repeatability 2 LRT – I/O Request Throughput Distribution Data

401 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	17:09:17	17:12:17	0-2	0:03:00
Measurement Interval	17:12:17	17:22:17	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	20,052.18	11,949.38	2,468.02	5,634.78
1	20,041.80	11,959.03	2,455.45	5,627.32
2	20,069.40	11,958.53	2,463.68	5,647.18
3	20,079.65	11,974.37	2,471.60	5,633.68
4	20,078.03	11,967.20	2,475.83	5,635.00
5	20,055.08	11,946.07	2,473.18	5,635.83
6	20,048.38	11,956.10	2,462.43	5,629.85
7	20,027.22	11,915.43	2,467.60	5,644.18
8	20,022.78	11,944.75	2,454.33	5,623.70
9	20,052.98	11,955.02	2,460.08	5,637.88
10	20,036.62	11,946.25	2,456.20	5,634.17
11	20,032.97	11,929.77	2,470.87	5,632.33
12	20,068.82	11,945.87	2,471.38	5,651.57
Average	***20,050.25***	***11,948.08***	***2,466.35***	***5,635.82***

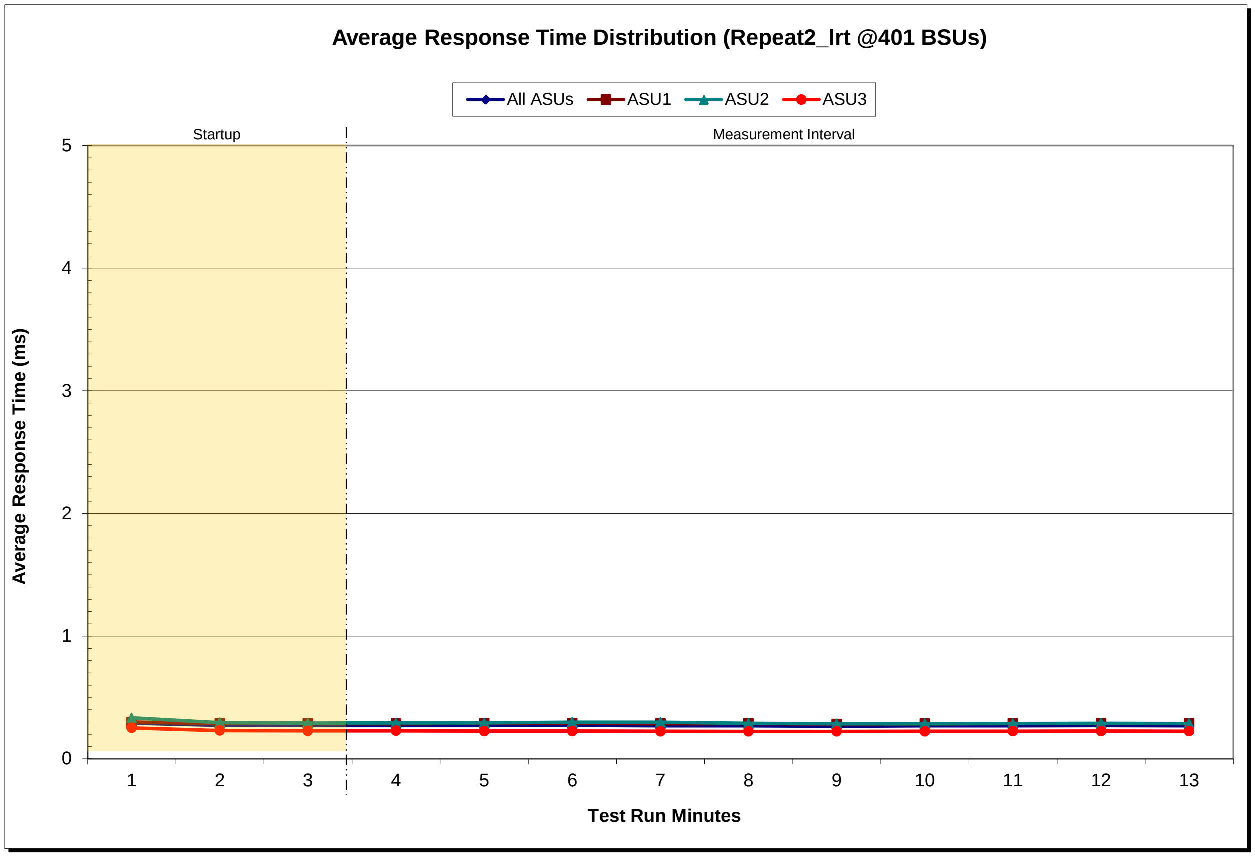
Repeatability 2 LRT – I/O Request Throughput Distribution Graph



Repeatability 2 LRT –Average Response Time (ms) Distribution Data

401 BSUs	Start	Stop	Interval	Duration
Start-Up/Ramp-Up	17:09:17	17:12:17	0-2	0:03:00
Measurement Interval	17:12:17	17:22:17	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	0.29	0.30	0.33	0.25
1	0.27	0.29	0.29	0.23
2	0.27	0.29	0.29	0.23
3	0.27	0.29	0.29	0.23
4	0.27	0.29	0.29	0.23
5	0.27	0.29	0.30	0.23
6	0.27	0.29	0.30	0.22
7	0.27	0.29	0.29	0.22
8	0.27	0.28	0.29	0.22
9	0.27	0.29	0.29	0.22
10	0.27	0.29	0.29	0.22
11	0.27	0.29	0.29	0.23
12	0.27	0.29	0.29	0.22
Average	***0.27***	***0.29***	***0.29***	***0.22***

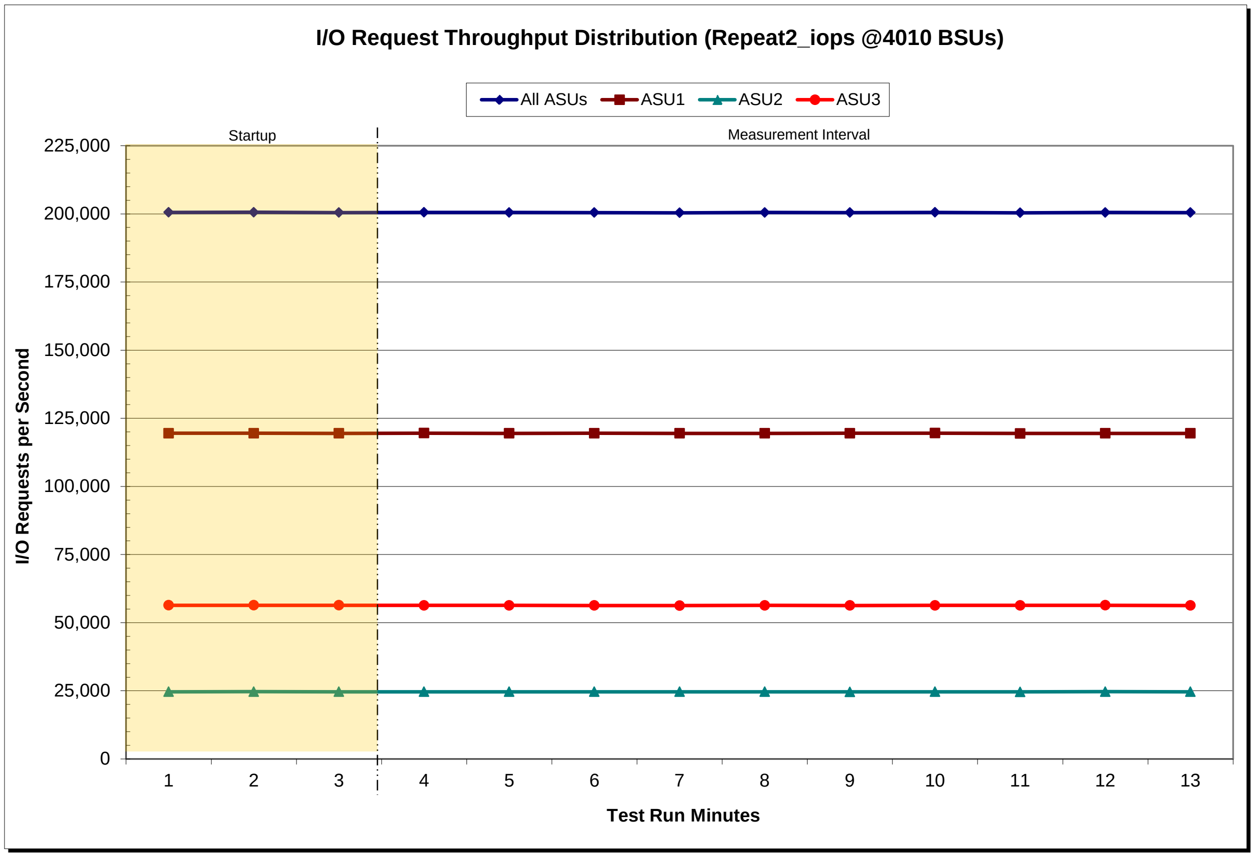
Repeatability 2 LRT –Average Response Time (ms) Distribution Graph



Repeatability 2 IOPS – I/O Request Throughput Distribution Data

4,010 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	17:23:44	17:26:44	0-2	0:03:00
Measurement Interval	17:26:44	17:36:44	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	200,567.63	119,494.13	24,668.15	56,405.35
1	200,576.22	119,500.57	24,681.68	56,393.97
2	200,487.85	119,466.80	24,635.27	56,385.78
3	200,546.30	119,541.13	24,663.43	56,341.73
4	200,495.60	119,487.57	24,644.27	56,363.77
5	200,489.55	119,519.08	24,645.77	56,324.70
6	200,405.13	119,483.37	24,637.75	56,284.02
7	200,497.53	119,469.77	24,658.67	56,369.10
8	200,461.17	119,513.20	24,627.00	56,320.97
9	200,553.05	119,543.08	24,652.95	56,357.02
10	200,401.22	119,415.85	24,632.43	56,352.93
11	200,531.25	119,451.13	24,679.10	56,401.02
12	200,452.97	119,472.95	24,673.95	56,306.07
Average	***200,483.38***	***119,489.71***	***24,651.53***	***56,342.13***

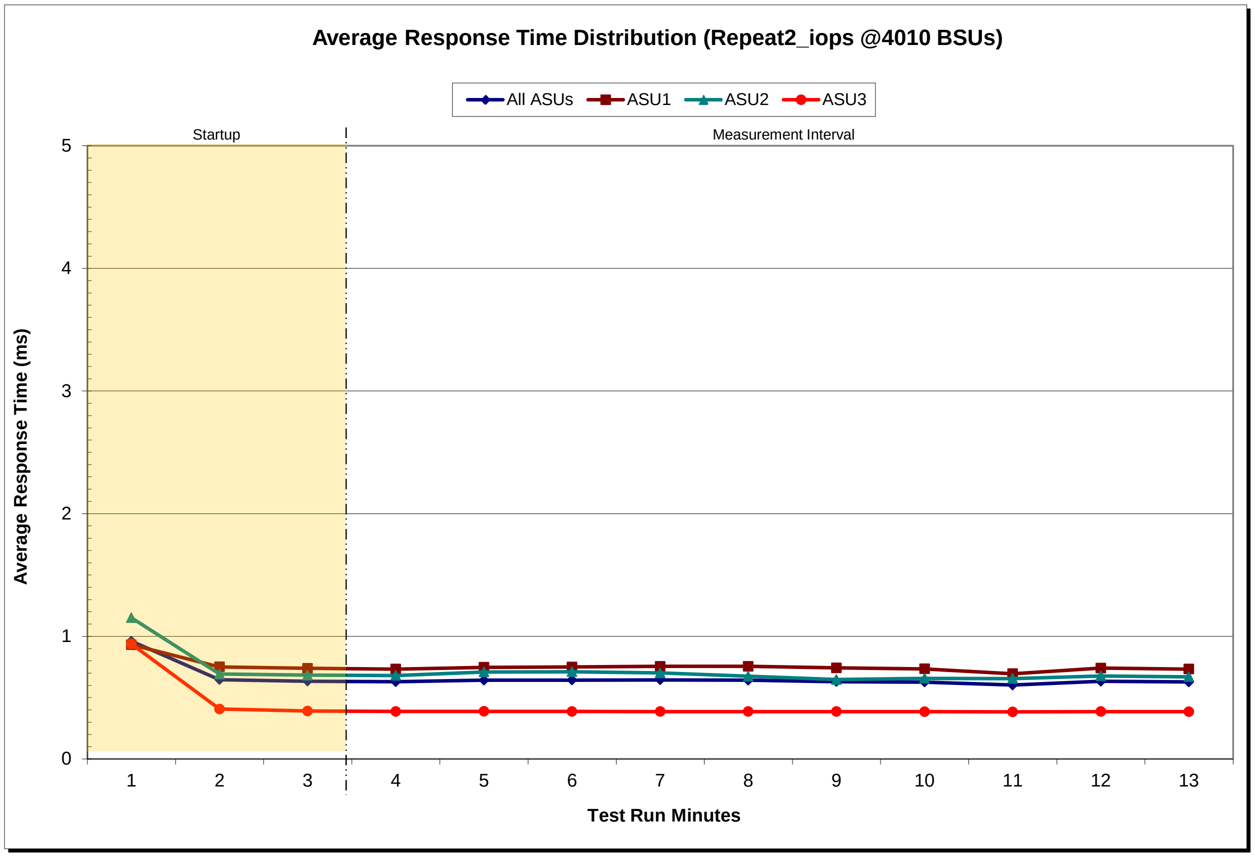
Repeatability 2 IOPS – I/O Request Throughput Distribution Graph



Repeatability 2 IOPS –Average Response Time (ms) Distribution Data

4,010 BSUs	**Start**	**Stop**	**Interval**	**Duration**
Start-Up/Ramp-Up	17:23:44	17:26:44	0-2	0:03:00
Measurement Interval	17:26:44	17:36:44	3-12	0:10:00
60 second intervals	**All ASUs**	**ASU1**	**ASU2**	**ASU3**
0	0.96	0.93	1.15	0.94
1	0.65	0.75	0.69	0.41
2	0.63	0.74	0.68	0.39
3	0.63	0.73	0.68	0.39
4	0.64	0.75	0.71	0.39
5	0.64	0.75	0.71	0.39
6	0.65	0.76	0.70	0.39
7	0.64	0.76	0.67	0.39
8	0.63	0.74	0.65	0.39
9	0.63	0.73	0.66	0.39
10	0.60	0.70	0.66	0.38
11	0.63	0.74	0.68	0.39
12	0.63	0.73	0.67	0.39
Average	***0.63***	***0.74***	***0.68***	***0.39***

Repeatability 2 IOPS –Average Response Time (ms) Distribution Graph



Repeatability 1 (LRT)
Measured Intensity Multiplier and Coefficient of Variation

<u>*Clause 3.4.3*</u>

IM – Intensity Multiplier: *The ratio of I/Os for each I/O stream relative to the total I/Os for all I/O streams (ASU1-1 – ASU3-1) as required by the benchmark specification.*

<u>*Clauses5.1.10 and 5.3.15.2*</u>

MIM – Measured Intensity Multiplier: *The Measured Intensity Multiplier represents the ratio of measured I/Os for each I/O stream relative to the total I/Os measured for all I/O streams (ASU1-1 – ASU3-1). This value may differ from the corresponding Expected Intensity Multiplier by no more than 5%.*

<u>*Clause 5.3.15.3*</u>

COV – Coefficient of Variation: *This measure of variation for the Measured Intensity Multiplier cannot exceed 0.2.*

	ASU1-1	**ASU1-2**	**ASU1-3**	**ASU1-4**	**ASU2-1**	**ASU2-2**	**ASU2-3**	**ASU3-1**
IM	***0.0350***	***0.2810***	***0.0700***	***0.2100***	***0.0180***	***0.0700***	***0.0350***	***0.2810***
MIM	0.0350	0.2809	0.0700	0.2100	0.0180	0.0701	0.0349	0.2810
COV	0.004	0.001	0.003	0.001	0.009	0.003	0.004	0.001

Repeatability 1 (IOPS)
Measured Intensity Multiplier and Coefficient of Variation

	ASU1-1	**ASU1-2**	**ASU1-3**	**ASU1-4**	**ASU2-1**	**ASU2-2**	**ASU2-3**	**ASU3-1**
IM	***0.0350***	***0.2810***	***0.0700***	***0.2100***	***0.0180***	***0.0700***	***0.0350***	***0.2810***
MIM	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
COV	0.001	0.001	0.001	0.000	0.003	0.001	0.001	0.001

Repeatability 2 (LRT)
Measured Intensity Multiplier and Coefficient of Variation

	ASU1-1	**ASU1-2**	**ASU1-3**	**ASU1-4**	**ASU2-1**	**ASU2-2**	**ASU2-3**	**ASU3-1**
IM	***0.0350***	***0.2810***	***0.0700***	***0.2100***	***0.0180***	***0.0700***	***0.0350***	***0.2810***
MIM	0.0349	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2811
COV	0.004	0.002	0.003	0.002	0.007	0.002	0.005	0.001

Repeatability 2 (IOPS)
Measured Intensity Multiplier and Coefficient of Variation

	ASU1-1	ASU1-2	ASU1-3	ASU1-4	ASU2-1	ASU2-2	ASU2-3	ASU3-1
IM	***0.0350***	***0.2810***	***0.0700***	***0.2100***	***0.0180***	***0.0700***	***0.0350***	***0.2810***
MIM	0.0350	0.2810	0.0700	0.2100	0.0180	0.0700	0.0350	0.2810
COV	0.002	0.000	0.001	0.001	0.002	0.001	0.001	0.000

Data Persistence Test

Clause 6

The Data Persistence Test demonstrates the Tested Storage Configuration (TSC):

- *Is capable of maintain data integrity across a power cycle.*
- *Ensures the transfer of data between Logical Volumes and host systems occurs without corruption or loss.*

The SPC-1 Workload Generator will write 16 block I/O requests at random over the total Addressable Storage Capacity of the TSC for ten (10) minutes at a minimum of 25% of the load used to generate the SPC-1 IOPS™ primary metric. The bit pattern selected to be written to each block as well as the address of the block will be retained in a log file.

The Tested Storage Configuration (TSC) will be shutdown and restarted using a power off/power on cycle at the end of the above sequence of write operations. In addition, any caches employing battery backup must be flushed/emptied.

The SPC-1 Workload Generator will then use the above log file to verify each block written contains the correct bit pattern.

Clause 9.4.3.8

The following content shall appear in this section of the FDR:

1. *A listing or screen image of all input parameters supplied to the Workload Generator.*
2. *For the successful Data Persistence Test Run, a table illustrating key results. The content, appearance, and format of this table are specified in Table 9-12. Information displayed in this table shall be obtained from the Test Run Results File referenced below in #3.*
3. *For the successful Data Persistence Test Run, the human readable Test Run Results file produced by the Workload Generator (may be contained in an appendix).*

SPC-1 Workload Generator Input Parameters

The SPC-1 Workload Generator input parameters for the Sustainability, IOPS, Response Time Ramp, Repeatability, and Persistence Test Runs are documented in Appendix E: SPC-1 Workload Generator Input Parameters on Page 76.

Data Persistence Test Results File

A link to each test result file generated from each Data Persistence Test is listed below.

Persistence 1 Test Results File

Persistence 2 Test Results File

Data Persistence Test Results

Data Persistence Test Results	
Data Persistence Test Run Number: 1	
Total Number of Logical Blocks Written	965,251
Total Number of Logical Blocks Verified	837,802
Total Number of Logical Blocks that Failed Verification	0
Time Duration for Writing Test Logical Blocks	5 minutes
Size in bytes of each Logical Block	1024
Number of Failed I/O Requests in the process of the Test	0

If approved by the SPC Auditor, the SPC-2 Persistence Test may be used to meet the SPC-1 persistence requirements. Both the SPC-1 and SPC-2 Persistence Tests provide the same level of functionality and verification of data integrity. The SPC-2 Persistence Test may be easily configured to address an SPC-1 storage configuration. The SPC-2 Persistence Test extends the size of storage configurations that may be tested and significantly reduces the test duration of such configurations.

The SPC-2 Persistence Test was approved for use in this set of audited measurements.

In some cases the same address was the target of multiple writes, which resulted in more Logical Blocks Written than Logical Blocks Verified. In the case of multiple writes to the same address, the pattern written and verified must be associated with the last write to that address.

PRICED STORAGE CONFIGURATION AVAILABILITY DATE

Clause 9.4.3.9

The committed delivery data for general availability (Availability Date) of all products that comprise the Priced Storage Configuration must be reported. When the Priced Storage Configuration includes products or components with different availability dates, the reported Availability Date for the Priced Storage Configuration must be the date at which all components are committed to be available.

The Fujitsu Storage Systems ETERNUS DX200 S3 as documented in this Full Disclosure Report is currently available for customer purchase and shipment.

PRICING INFORMATION

Clause 9.4.3.3.6

The Executive Summary shall contain a pricing spreadsheet as documented in Clause 8.3.1.

Pricing information may be found in the Priced Storage Configuration Pricing section on page 18.

TESTED STORAGE CONFIGURATION (TSC) AND PRICED STORAGE CONFIGURATION DIFFERENCES

Clause 9.4.3.3.8

The Executive Summary shall contain a list of all differences between the Tested Storage Configuration (TSC) and the Priced Storage Configuration.

A list of all differences between the Tested Storage Configuration (TSC) and Priced Storage Configuration may be found in the Executive Summary portion of this document on page 18.

ANOMALIES OR IRREGULARITIES

Clause 9.4.3.10

The FDR shall include a clear and complete description of any anomalies or irregularities encountered in the course of executing the SPC-1 benchmark that may in any way call into question the accuracy, verifiability, or authenticity of information published in this FDR.

There were no anomalies or irregularities encountered during the SPC-1 Remote Audit of the Fujitsu Storage Systems ETERNUS DX200 S3.

APPENDIX A: SPC-1 GLOSSARY

"Decimal" *(powers of ten)* Measurement Units

In the storage industry, the terms "kilo", "mega", "giga", "tera", "peta", and "exa" are commonly used prefixes for computing performance and capacity. For the purposes of the SPC workload definitions, all of the following terms are defined in "powers of ten" measurement units.

A kilobyte (KB) is equal to 1,000 (10^3) bytes.

A megabyte (MB) is equal to 1,000,000 (10^6) bytes.

A gigabyte (GB) is equal to 1,000,000,000 (10^9) bytes.

A terabyte (TB) is equal to 1,000,000,000,000 (10^{12}) bytes.

A petabyte (PB) is equal to 1,000,000,000,000,000 (10^{15}) bytes

An exabyte (EB) is equal to 1,000,000,000,000,000,000 (10^{18}) bytes

"Binary" *(powers of two)* Measurement Units

The sizes reported by many operating system components use "powers of two" measurement units rather than "power of ten" units. The following standardized definitions and terms are also valid and may be used in this document.

A kibibyte (KiB) is equal to 1,024 (2^{10}) bytes.

A mebibyte (MiB) is equal to 1,048,576 (2^{20}) bytes.

A gigibyte (GiB) is equal to 1,073,741,824 (2^{30}) bytes.

A tebibyte (TiB) is equal to 1,099,511,627,776 (2^{40}) bytes.

A pebibyte (PiB) is equal to 1,125,899,906,842,624 (2^{50}) bytes.

An exbibyte (EiB) is equal to 1,152,921,504,606,846,967 (2^{60}) bytes.

SPC-1 Data Repository Definitions

Total ASU Capacity: The total storage capacity read and written in the course of executing the SPC-1 benchmark.

Application Storage Unit (ASU): The logical interface between the storage and SPC-1 Workload Generator. The three ASUs (Data, User, and Log) are typically implemented on one or more Logical Volume.

Logical Volume: The division of Addressable Storage Capacity into individually addressable logical units of storage used in the SPC-1 benchmark. Each Logical Volume is implemented as a single, contiguous address space.

Addressable Storage Capacity: The total storage (sum of Logical Volumes) that can be read and written by application programs such as the SPC-1 Workload Generator.

Configured Storage Capacity: This capacity includes the Addressable Storage Capacity and any other storage (parity disks, hot spares, etc.) necessary to implement the Addressable Storage Capacity.

Physical Storage Capacity: The formatted capacity of all storage devices physically present in the Tested Storage Configuration (TSC).

Data Protection Overhead: The storage capacity required to implement the selected level of data protection.

Required Storage: The amount of Configured Storage Capacity required to implement the Addressable Storage Configuration, excluding the storage required for the three ASUs.

Global Storage Overhead: The amount of Physical Storage Capacity that is required for storage subsystem use and unavailable for use by application programs.

Total Unused Storage: The amount of storage capacity available for use by application programs but not included in the Total ASU Capacity.

SPC-1 Data Protection Levels

Protected 1: The single point of failure of any ***storage device*** in the configuration will not result in permanent loss of access to or integrity of the SPC-1 Data Repository.

Protected 2: The single point of failure of any ***component*** in the configuration will not result in permanent loss of access to or integrity of the SPC-1 Data Repository.

SPC-1 Test Execution Definitions

Average Response Time: The sum of the Response Times for all Measured I/O Requests divided by the total number of Measured I/O Requests.

Completed I/O Request: An I/O Request with a Start Time and a Completion Time (see "I/O Completion Types" below).

Completion Time: The time recorded by the Workload Generator when an I/O Request is satisfied by the TSC as signaled by System Software.

Data Rate: The data transferred in all Measured I/O Requests in an SPC-1 Test Run divided by the length of the Test Run in seconds.

Expected I/O Count: For any given I/O Stream and Test Phase, the product of 50 times the BSU level, the duration of the Test Phase in seconds, and the Intensity Multiplier for that I/O Stream.

Failed I/O Request: Any I/O Request issued by the Workload Generator that could not be completed or was signaled as failed by System Software. A Failed I/O Request has no Completion Time (see "I/O Completion Types" below).

I/O Request Throughput: The total number of Measured I/O requests in an SPC-1 Test Run divided by the duration of the Measurement Interval in seconds.

In-Flight I/O Request: An I/O Request issued by the I/O Command Generator to the TSC that has a recorded Start Time, but does not complete within the Measurement Interval (see "I/O Completion Types" below).

Measured I/O Request: A Completed I/O Request with a Completion Time occurring within the Measurement Interval (see "I/O Completion Types" below).

Measured Intensity Multiplier: The percentage of all Measured I/O Requests that were issued by a given I/O Stream.

Measurement Interval: The finite and contiguous time period, after the TSC has reached Steady State, when data is collected by a Test Sponsor to generate an SPC-1 test result or support an SPC-1 test result.

Ramp-Up: The time required for the Benchmark Configuration (BC) to produce Steady State throughput after the Workload Generator begins submitting I/O Requests to the TSC for execution.

Ramp-Down: The time required for the BC to complete all I/O Requests issued by the Workload Generator. The Ramp-Down period begins when the Workload Generator ceases to issue new I/O Requests to the TSC.

Response Time: The Response Time of a Measured I/O Request is its Completion Time minus its Start Time.

Start Time: The time recorded by the Workload Generator when an I/O Request is submitted, by the Workload Generator, to the System Software for execution on the Tested Storage Configuration (TSC).

Start-Up: The period that begins after the Workload Generator starts to submit I/O requests to the TSC and ends at the beginning of the Measurement Interval.

Shut-Down: The period between the end of the Measurement Interval and the time when all I/O Requests issued by the Workload Generator have completed or failed.

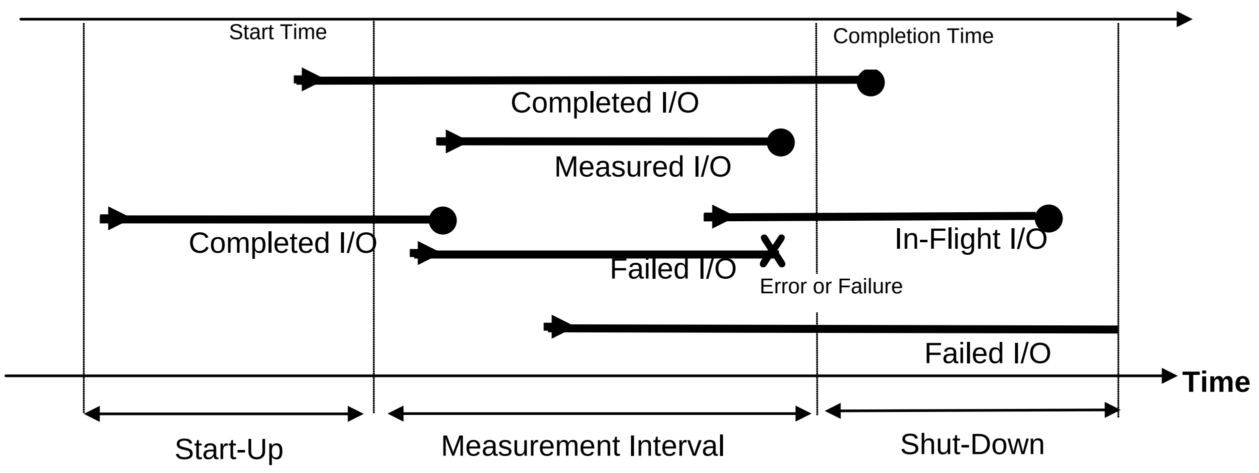
Steady State: The consistent and sustainable throughput of the TSC. During this period the load presented to the TSC by the Workload Generator is constant.

Test: A collection of Test Phases and or Test Runs sharing a common objective.

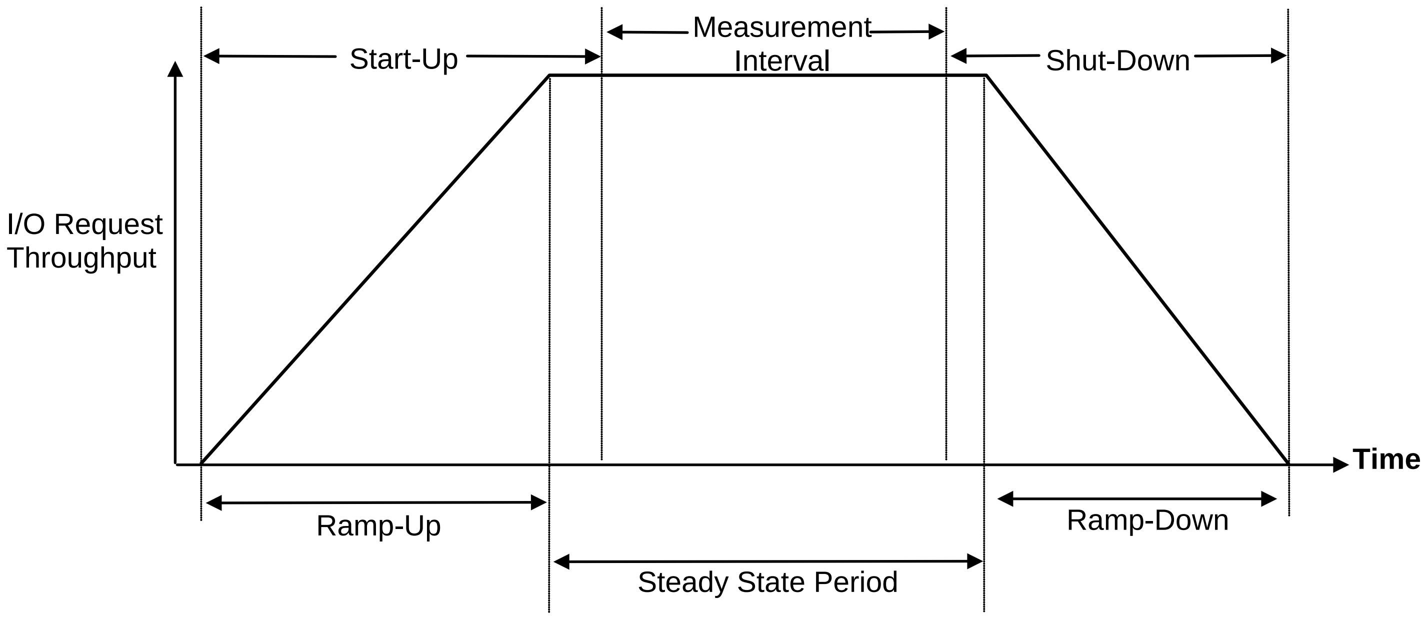
Test Run: The execution of SPC-1 for the purpose of producing or supporting an SPC-1 test result. SPC-1 Test Runs may have a finite and measured Ramp-Up period, Start-Up period, Shut-Down period, and Ramp-Down period as illustrated in the "SPC-1 Test Run Components" below. All SPC-1 Test Runs shall have a Steady State period and a Measurement Interval.

Test Phase: A collection of one or more SPC-1 Test Runs sharing a common objective and intended to be run in a specific sequence.

I/O Completion Types



SPC-1 Test Run Components



APPENDIX B: CUSTOMER TUNABLE PARAMETERS AND OPTIONS

Copy the **qla2xxx.conf** file to **/etc/modprobe.d/** on each Host System and reboot. This parameter change sets the maximum queue depth to 128 from the default of 32.

qla2xxx.conf

```
options qla2xxx ql2xmaxqdepth=128
```

APPENDIX C: TESTED STORAGE CONFIGURATION (TSC) CREATION

The standard Fujitsu Command Line tool (CLI) was used to create the ETERNUS DX200 S3 SPC-1 configuration.

The 'master' script, **doFDRcfg.sh**, was executed, which in turn, invoked the script, **DX200S3_20131015.exp**. The 'master' script included shell commands to monitor the progress as the physical formatting proceeded, which used the **expect** script **showFormatStatus.exp** to pick up the status information from the array.

The **DX200S3_20131015.exp** script completed steps 1-4, described below for the 8 host port configuration.

Each **expect** script included the **docli** procedure, which was used to issue the CLI commands to the array. That procedure used **ssh** for communication with the array. A second procedure in the script, **doexit**, was used to conclude the execution sequence at the end of the script.

Step 1 – Creation of RAID Groups

A total of 14 RAID Groups were created, according to the configuration plan, **DX200S3_20131015_Configuration.xlsx**, which is typically prepared in concert with a Fujitsu SE. Each RAID Group was made up of 2 disk drives in a RAID1 (1+1) configuration and assigned to a specific CM for operational control. The RAID Groups were named RG00 through RG13.

Step 2 – Creation of the Logical Volumes

Wide striped logical volumes were created across sets of 7 RAID Groups. Two volumes were created on RAID Groups owned by each CM for each of the three ASUs, for a total of 12 logical volumes. The sizes of the volumes created for ASU-1 and ASU-2 were set to 589,824 MiB (618,475.291 MB). The sizes of the volumes created for ASU-3 were set to 131,031 MiB (137,395.962 MB).

Step 3 – Creation of the Global Hot Spares

One drive was designated as the Global Hot Spare in slot 14 of the CE, per the configuration plan.

Step 4 – Assignment of LUN Mapping to the Linux Host Systems

The **DX200S3_20131015.exp** script provided mapping to eight host ports.

The port LUN mapping was assigned for each of the Logical Volumes using two ports on each of the two Channel Adapters (CA) in each of the two Controller Modules (CM). Each of the even numbered volumes, which were defined on RAID Groups owned by CM-0, were assigned LUN numbers on the four active ports on the two CAs installed on CM-0. Each of the odd numbered volumes, which were defined on RAID Groups owned by CM-1, were assigned LUN numbers on the four active ports on the two CAs installed on CM-1.

TSC Creation/Configuration Scripts

doFDRcfg.sh

```
#!/bin/bash
#
# Do the configuration steps required for the SPC1 benchmark
#
# create tmp directory for spc1 if it does not exist
if [ ! -d /tmp/spc1 ]; then
mkdir /tmp/spc1
fi
ROOT=/home/spc1/fdr/fdrwork
SCRIPTS=${ROOT}/7_Execution
CONFIGURE=${ROOT}/5_Creation
#
# confID uniquely identifies the configuration of the array
confID=DX200S3_20131015
#
# obtain cjobID based on the timestamp
# cjobID uniquely identifies the configuration job
cjobID=C`date +%y%m%d%H%M%S`
#
echo job start time `date` > /tmp/spc1/${cjobID}_message.txt
echo This is an array configuration job >> /tmp/spc1/${cjobID}_message.txt
echo job confID=$confID >> /tmp/spc1/${cjobID}_message.txt
echo job cjobID=$cjobID >> /tmp/spc1/${cjobID}_message.txt
${SCRIPTS}/sendstatus.sh "Starting Configuration Job=${cjobID}"
${cjobID}_message.txt
#
# Configure Array using the Expect script to issue CLI commands
#
${SCRIPTS}/sendstatus.sh "Starting Eternus CLI script for configuration
Job=${cjobID}" ${cjobID}_message.txt
${CONFIGURE}/${confID}.exp
${SCRIPTS}/sendstatus.sh "Completed Eternus CLI script for configuration
Job=${cjobID}" ${cjobID}_message.txt
#
#
# Wait for physical format to complete
#
PollingInterval=1200 #wait 20 minutes to check format status
${SCRIPTS}/sendstatus.sh "Waiting for physical format to complete Job=${cjobID}"
${cjobID}_message.txt
#
${CONFIGURE}/showFormatStatus.exp dx200s3 root /tmp/spc1/fmt_${cjobID}.txt
#
LUNS=`grep Available /tmp/spc1/fmt_${cjobID}.txt|wc|awk '{print $1}'`
while [ $LUNS -gt 0 ]; do
 echo "-------------------------------------------"  >>
/tmp/spc1/${cjobID}_message.txt
 cat  /tmp/spc1/fmt_${cjobID}.txt >> /tmp/spc1/${cjobID}_message.txt
 ${SCRIPTS}/sendstatus.sh "Currently formatting $LUNS LUNS Job=${cjobID}"
${cjobID}_message.txt
 sleep $PollingInterval
 ${CONFIGURE}/showFormatStatus.exp dx200s3 root /tmp/spc1/fmt_${cjobID}.txt
 LUNS=`grep Available /tmp/spc1/fmt_${cjobID}.txt |wc |awk '{print $1}'`
done
${SCRIPTS}/sendstatus.sh "Physical format complete please proceed. Job=${cjobID}"
${cjobID}_message.txt
```

DX200S3_20131015.exp

```
#!/usr/bin/expect
# script to setup initial configuration for dx200s3
# for SPC-1 benchmark
# Requirment: ssh public key for this server  registered to the array
set timeout 600
set user root
spawn ssh dx200s3 -l $user
expect "CLI>"
# procedure to execute dx200s3 cli command
proc docli { cmd args} {
send "$cmd $args
expect "CLI>"
}
# procedure to exit
proc doexit {} {
send "exit
}

#DX200S3 Configuration

## Create RAID ##
docli create raid-group -name RG00 -disks 0000,0100 -level 1 -assigned-cm 0
docli create raid-group -name RG01 -disks 0001,0101 -level 1 -assigned-cm 1
docli create raid-group -name RG02 -disks 0002,0102 -level 1 -assigned-cm 0
docli create raid-group -name RG03 -disks 0003,0103 -level 1 -assigned-cm 1
docli create raid-group -name RG04 -disks 0004,0104 -level 1 -assigned-cm 0
docli create raid-group -name RG05 -disks 0005,0105 -level 1 -assigned-cm 1
docli create raid-group -name RG06 -disks 0006,0106 -level 1 -assigned-cm 0
docli create raid-group -name RG07 -disks 0007,0107 -level 1 -assigned-cm 1
docli create raid-group -name RG08 -disks 0008,0108 -level 1 -assigned-cm 0
docli create raid-group -name RG09 -disks 0009,0109 -level 1 -assigned-cm 1
docli create raid-group -name RG10 -disks 0010,0110 -level 1 -assigned-cm 0
docli create raid-group -name RG11 -disks 0011,0111 -level 1 -assigned-cm 1
docli create raid-group -name RG12 -disks 0012,0112 -level 1 -assigned-cm 0
docli create raid-group -name RG13 -disks 0013,0113 -level 1 -assigned-cm 1

## Create Volume ##
docli create volume -name ASU1-1 -count 1 -rg-name
RG00,RG02,RG04,RG06,RG08,RG10,RG12 -type wsv -size 589824mb
docli create volume -name ASU1-2 -count 1 -rg-name
RG01,RG03,RG05,RG07,RG09,RG11,RG13 -type wsv -size 589824mb
docli create volume -name ASU1-3 -count 1 -rg-name
RG00,RG02,RG04,RG06,RG08,RG10,RG12 -type wsv -size 589824mb
docli create volume -name ASU1-4 -count 1 -rg-name
RG01,RG03,RG05,RG07,RG09,RG11,RG13 -type wsv -size 589824mb
docli create volume -name ASU2-1 -count 1 -rg-name
RG00,RG02,RG04,RG06,RG08,RG10,RG12 -type wsv -size 589824mb
docli create volume -name ASU2-2 -count 1 -rg-name
RG01,RG03,RG05,RG07,RG09,RG11,RG13 -type wsv -size 589824mb
docli create volume -name ASU2-3 -count 1 -rg-name
RG00,RG02,RG04,RG06,RG08,RG10,RG12 -type wsv -size 589824mb
docli create volume -name ASU2-4 -count 1 -rg-name
RG01,RG03,RG05,RG07,RG09,RG11,RG13 -type wsv -size 589824mb
docli create volume -name ASU3-1 -count 1 -rg-name
RG00,RG02,RG04,RG06,RG08,RG10,RG12 -type wsv -size 131031mb
docli create volume -name ASU3-2 -count 1 -rg-name
RG01,RG03,RG05,RG07,RG09,RG11,RG13 -type wsv -size 131031mb
docli create volume -name ASU3-3 -count 1 -rg-name
RG00,RG02,RG04,RG06,RG08,RG10,RG12 -type wsv -size 131031mb
docli create volume -name ASU3-4 -count 1 -rg-name
RG01,RG03,RG05,RG07,RG09,RG11,RG13 -type wsv -size 131031mb
```

```
## Set Global Hot Spare ##
docli set global-spare -disks 0014

## Set LUN Mapping ##
docli set mapping -port 000,001,010,011 -volume-number 0,2,4,6,8,10 -lun
0,2,4,6,8,10
docli set mapping -port 100,101,110,111 -volume-number 1,3,5,7,9,11 -lun
1,3,5,7,9,11

## Logout ##
doexit
```

showFormatStatus.exp

```
#!/usr/bin/expect -f
# Create volumes from the array
# getFormatStatus  <array> <arrayid> <file>
# assumption: array's ssh port has ssh-key-pre-registered no no password is required
# please register ssh-keys
# procedure to execute commands
proc docli {cmd args} {
send "$cmd $args\r"
expect "CLI>"
}
# procedure to exit
proc doexit {} {
send "exit \r"
}
set array [lindex $argv 0]
set arrayid [lindex $argv 1]
set file [lindex $argv 2]
#set file /tmp/formatstatus.txt
# login
spawn ssh $arrayid@$array
set timeout 40
expect "CLI>"
if [catch {open $file "w" } output] {
  puts "$output"
  exit
}
send "show volume-progress\r"
expect "CLI>"
puts $output "Output = $expect_out(buffer)"
close $output
doexit
close
```

APPENDIX D: SPC-1 WORKLOAD GENERATOR STORAGE COMMANDS AND PARAMETERS

ASU Pre-Fill

The content of command and parameter file, used in this benchmark to execute the required ASU pre-fill, is listed below.

```
*
* Prefill vdbench parameter file for SPC1 DX200S3 2013/10/15
*
* This will produce a random data pattern of the entire LBA range using LSFR
* 32 bit
*
compratio=1
*
*
*  openflags=directio is specified for linux since vdbench requires this when /dev
is used
*  size parameter is also specified because linux version of vdbench requires it
sd=asu1_1,lun=/dev/raw/raw100,size=369g,threads=32,openflags=directio *sdnum=1
sd=asu1_2,lun=/dev/raw/raw101,size=369g,threads=32,openflags=directio *sdnum=2
sd=asu1_3,lun=/dev/raw/raw203,size=369g,threads=32,openflags=directio *sdnum=3
sd=asu1_4,lun=/dev/raw/raw202,size=369g,threads=32,openflags=directio *sdnum=4
sd=asu2_1,lun=/dev/raw/raw104,size=369g,threads=32,openflags=directio *sdnum=5
sd=asu2_2,lun=/dev/raw/raw105,size=369g,threads=32,openflags=directio *sdnum=6
sd=asu2_3,lun=/dev/raw/raw207,size=369g,threads=32,openflags=directio *sdnum=7
sd=asu2_4,lun=/dev/raw/raw206,size=369g,threads=32,openflags=directio *sdnum=8
sd=asu3_1,lun=/dev/raw/raw109,size=82g,threads=32,openflags=directio *sdnum=9
sd=asu3_2,lun=/dev/raw/raw108,size=82g,threads=32,openflags=directio *sdnum=10
sd=asu3_3,lun=/dev/raw/raw210,size=82g,threads=32,openflags=directio *sdnum=11
sd=asu3_4,lun=/dev/raw/raw211,size=82g,threads=32,openflags=directio *sdnum=12
*
wd=wd1,sd=asu1_1,rdpct=0,seek=-1,xfersize=128K
wd=wd2,sd=asu1_2,rdpct=0,seek=-1,xfersize=128K
wd=wd3,sd=asu1_3,rdpct=0,seek=-1,xfersize=128K
wd=wd4,sd=asu1_4,rdpct=0,seek=-1,xfersize=128K
wd=wd5,sd=asu2_1,rdpct=0,seek=-1,xfersize=128K
wd=wd6,sd=asu2_2,rdpct=0,seek=-1,xfersize=128K
wd=wd7,sd=asu2_3,rdpct=0,seek=-1,xfersize=128K
wd=wd8,sd=asu2_4,rdpct=0,seek=-1,xfersize=128K
wd=wd9,sd=asu3_1,rdpct=0,seek=-1,xfersize=128K
wd=wd10,sd=asu3_2,rdpct=0,seek=-1,xfersize=128K
wd=wd11,sd=asu3_3,rdpct=0,seek=-1,xfersize=128K
wd=wd12,sd=asu3_4,rdpct=0,seek=-1,xfersize=128K
*
*===================================
* Use 10 hours as a maximum elapsed time,
* which should ensure the entire LBA range
* will be written before the time elapses
*===================================
rd=asu_prefill,wd=wd*,iorate=max,elapsed=36000,interval=10
* The above "elapsed=36000" may have to be increased to ensure that the utility will
reach
* the end of the LUN ("seek=-1") prior to the end of the specified elapsed time
```

Common Command Lines – Primary Metrics and Repeatability Tests

The following command lines appear at the beginning of each command and parameter file for the Primary Metrics and Repeatability Test. The command lines are only listed below to eliminate redundancy.

```
host=master
slaves=(slave1,slave2,slave3,slave4,slave5,slave6,slave7,slave8,slave9,slave10,slave
11,slave12,slave13,slave14,slave15,slave16,slave17,slave18,slave19,slave20,slave21,s
lave22,slave23,slave24,slave25,slave26,slave27,slave28,slave29,slave30,slave31,slave
32,slave33,slave34,slave35,slave36,slave37,slave38,slave39,slave40,slave41,slave42,s
lave43,slave44,slave45,slave46,slave47,slave48,slave49,slave50)
javaparms="-Xmx3072m -Xms3072m -Xss512k"
sd=asu1_1,lun=/dev/raw/raw100,size=369g #sdnum=1
sd=asu1_2,lun=/dev/raw/raw101,size=369g #sdnum=2
sd=asu1_3,lun=/dev/raw/raw203,size=369g #sdnum=3
sd=asu1_4,lun=/dev/raw/raw202,size=369g #sdnum=4
sd=asu2_1,lun=/dev/raw/raw104,size=369g #sdnum=5
sd=asu2_2,lun=/dev/raw/raw105,size=369g #sdnum=6
sd=asu2_3,lun=/dev/raw/raw207,size=369g #sdnum=7
sd=asu2_4,lun=/dev/raw/raw206,size=369g #sdnum=8
sd=asu3_1,lun=/dev/raw/raw109,size=82g #sdnum=9
sd=asu3_2,lun=/dev/raw/raw108,size=82g #sdnum=10
sd=asu3_3,lun=/dev/raw/raw210,size=82g #sdnum=11
sd=asu3_4,lun=/dev/raw/raw211,size=82g #sdnum=12
```

Primary Metrics Test: Sustainability Test Phase/Test Run

common commands 1

```
rd=sustain,bsus=4010,startup=180,elapsed=28800,interval=60
```

Primary Metrics Test: IOPS Test Phase *(100% Test Run)*

common commands 1

```
rd=ramp_100,bsus=4010,startup=180,elapsed=600,interval=60
```

Primary Metrics Test: Response Time Ramp Test Phase *(95% Test Run)*

common commands 1

```
rd=ramp_95,bsus=3809,startup=180,elapsed=600,interval=60
```

Primary Metrics Test: Response Time Ramp Test Phase *(90% Test Run)*

common commands 1

```
rd=ramp_90,bsus=3609,startup=180,elapsed=600,interval=60
```

Primary Metrics Test: Response Time Ramp Test Phase *(80% Test Run)*

common commands 1

```
rd=ramp_80,bsus=3208,startup=180,elapsed=600,interval=60
```

Primary Metrics Test: Response Time Ramp Test Phase *(50% Test Run)*

common commands 1

```
rd=ramp_50,bsus=2005,startup=180,elapsed=600,interval=60
```

Primary Metrics Test: Response Time Ramp Test Phase *(10% Test Run)*

common commands 1

```
rd=ramp_10,bsus=401,startup=180,elapsed=600,interval=60
```

Repeatability Test: Repeatability Test Phase 1 (10% Test Run)

common commands 1

```
rd=repeat1_lrt,bsus=401,startup=180,elapsed=600,interval=60
```

Repeatability Test: Repeatability Test Phase 1 *(100% Test Run)*

common commands 1

```
rd=repeat1_iops,bsus=4010,startup=180,elapsed=600,interval=60
```

Repeatability Test: Repeatability Test Phase 2 *(10% Test Run)*

common commands 1

```
rd=repeat2_lrt,bsus=401,startup=180,elapsed=600,interval=60
```

Repeatability Test: Repeatability Test Phase 2 *(100% Test Run)*

common commands 1

```
rd=repeat2_iops,bsus=4010,startup=180,elapsed=600,interval=60
```

SPC-1 Persistence Test Run 1

The content of SPC-1 Workload Generator command and parameter file, used in this benchmark to execute a reduced level SPC-1 Persistence Test Run 1, is listed below.

```
javaparms="-Xmx3072m -Xms3072m -Xss512k"
sd=asu1_1,lun=/dev/raw/raw100,size=369g #sdnum=1
sd=asu1_2,lun=/dev/raw/raw101,size=369g #sdnum=2
sd=asu1_3,lun=/dev/raw/raw203,size=369g #sdnum=3
sd=asu1_4,lun=/dev/raw/raw202,size=369g #sdnum=4
sd=asu2_1,lun=/dev/raw/raw104,size=369g #sdnum=5
sd=asu2_2,lun=/dev/raw/raw105,size=369g #sdnum=6
sd=asu2_3,lun=/dev/raw/raw207,size=369g #sdnum=7
sd=asu2_4,lun=/dev/raw/raw206,size=369g #sdnum=8
sd=asu3_1,lun=/dev/raw/raw109,size=82g #sdnum=9
sd=asu3_2,lun=/dev/raw/raw108,size=82g #sdnum=10
sd=asu3_3,lun=/dev/raw/raw210,size=82g #sdnum=11
sd=asu3_4,lun=/dev/raw/raw211,size=82g #sdnum=12
```

SPC-2 Persistence Test

If approved by the SPC Auditor, the SPC-2 Persistence Test may be used to meet the SPC-1 persistence requirements. Both the SPC-1 and SPC-2 Persistence Tests provide the same level of functionality and verification of data integrity.

Common Command Lines – SPC-2 Persistence Test

The following command lines appear at the beginning of each command and parameter file for the two SPC-2 Persistence Test Runs. The command lines are only listed below to eliminate redundancy

```
host=localhost,jvms=2,java=("java","-Xmx3072m -Xms3072m -Xss512k")


sd=asu1_1,lun=/dev/raw/raw100,size=369g #sdnum=1
sd=asu1_2,lun=/dev/raw/raw101,size=369g #sdnum=2
sd=asu1_3,lun=/dev/raw/raw203,size=369g #sdnum=3
sd=asu1_4,lun=/dev/raw/raw202,size=369g #sdnum=4
sd=asu2_1,lun=/dev/raw/raw104,size=369g #sdnum=5
sd=asu2_2,lun=/dev/raw/raw105,size=369g #sdnum=6
sd=asu2_3,lun=/dev/raw/raw207,size=369g #sdnum=7
sd=asu2_4,lun=/dev/raw/raw206,size=369g #sdnum=8
sd=asu3_1,lun=/dev/raw/raw109,size=82g #sdnum=9
sd=asu3_2,lun=/dev/raw/raw108,size=82g #sdnum=10
sd=asu3_3,lun=/dev/raw/raw210,size=82g #sdnum=11
sd=asu3_4,lun=/dev/raw/raw211,size=82g #sdnum=12

maxlatestart=1
reportinginterval=5
segmentlength=512m
```

SPC-2 Persistence Test Run 1 *(write phase)*

```
* Persistence Test - Write Phase
```
common commands 2

```
rd=default,rampup=180,periods=90,measurement=300,runout=0,rampdown=0
rd=default,buffers=1,rdpct=0,xfersize=1024k

rd=TR1_SPC-2-persist-w,streams=134
```

SPC-2 Persistence Test Run 2 *(read phase)*

```
* Persistence Test - Read Phase
```
common commands 2

```
maxpersistenceerrors=10

rd=default,buffers=1,rdpct=100,xfersize=1024k

rd=TR1_SPC-2-persist-r
```

Slave JVMs

Each Slave JVM was invoked with a command and parameter file similar to the example listed below. The only difference in each file was **host** parameter value, which was unique to each Slave JVM, e.g. **slave1**…**slave50**.

```
master=masterh
host=slave1
javaparms="-Xmx3072m -Xms3072m -Xss512k"
sd=asu1_1,lun=/dev/raw/raw100,size=369g #sdnum=1
sd=asu1_2,lun=/dev/raw/raw101,size=369g #sdnum=2
sd=asu1_3,lun=/dev/raw/raw203,size=369g #sdnum=3
sd=asu1_4,lun=/dev/raw/raw202,size=369g #sdnum=4
sd=asu2_1,lun=/dev/raw/raw104,size=369g #sdnum=5
sd=asu2_2,lun=/dev/raw/raw105,size=369g #sdnum=6
sd=asu2_3,lun=/dev/raw/raw207,size=369g #sdnum=7
sd=asu2_4,lun=/dev/raw/raw206,size=369g #sdnum=8
sd=asu3_1,lun=/dev/raw/raw109,size=82g #sdnum=9
sd=asu3_2,lun=/dev/raw/raw108,size=82g #sdnum=10
sd=asu3_3,lun=/dev/raw/raw210,size=82g #sdnum=11
sd=asu3_4,lun=/dev/raw/raw211,size=82g #sdnum=12
```

APPENDIX E: SPC-1 WORKLOAD GENERATOR INPUT PARAMETERS

'Master' Execution Script

The 'master' script, **doFDRall_1M.sh**, was used to execute the required ASU pre-fill, Primary Metrics Test *(Sustainability Test Phase, IOPS Test Phase, and Response Time Ramp Test Phase)*, Repeatability Test *(Repeatability Test Phase 1 and Repeatability Test Phase 2)*, a reduced level SPC-1 Persistence Test Run 1 and SPC-2 Persistence Test Run 1 in an uninterrupted sequence.

After the above test sequence completed, the required TSC power off/power was performed. That step was followed by execution of the **doFDRall_2M.sh** script, which was used to perform SPC-2 Persistence Test Run 2.

The **doFDRall_1M.sh** and **doFDRall_2M.sh** scripts invoked various other scripts which appear below in the **Referenced Scripts** section with a brief description of each referenced script.

doFDRall_1M.sh

```
#!/bin/bash -x
#
# Script consists of Part 1 of the FDR job
#  - Save "Before Prefill" logs
#  - Prefill
#  - Save "Before FDR" logs
#  - Metricss
#  - Repeatability1/2
#  - Persistence1
# This script incorporates lowlevel scripts by Walter Baker.
#
#
if [ $# -ne 2 ]
then
 BSU=4010
 BSUPERSIST=2500
else
 BSU=$1
 BSUPERSIST=$2
fi
# Absolute path of Work Directories
ROOT=/home/spc1/fdr/fdrwork;export ROOT
# directory for prefill step
PREFILL=${ROOT}/prefill;export PREFILL
# main script directory
SCRIPTS=${ROOT}/7_Execution;export SCRIPTS
NSLAVES=50;export NSLAVES
# benchmark commands
#JAVA=${SCRIPTS}/javadummy.sh
JAVA=java
export JAVA
#VDBENCH=${SCRIPTS}/vdbenchdummy.sh
VDBENCH=vdbench
export VDBENCH
#EXPORTLOG=exportLogDummy.exp
EXPORTLOG=exportLog.exp
export EXPORTLOG
export NSLAVES
JAVAPARMS="-Xmx3072m -Xms3072m -Xss512k"
```

```
export JAVAPARMS
#Repository Info
BMTPATH=/usr/local/share3/StoragePerformance/SPC1_Benchmark/DX200S3/
export BMTPATH
BMTHOST=fjuser@129.212.198.24
export BMTHOST
#
# create tmp directory for spc1 if it does not exist
SPCTMP=/tmp/spc1
export SPCTMP
if [ ! -d ${SPCTMP} ]; then
mkdir ${SPCTMP}
fi
# confID uniquely identifies the configuration of the array
confID=DX200S3_20131015
export confID
# obtain jobID based on the timestamp
# jobID uniquely identifies the benchmark run
jobID=J`date +%y%m%d%H%M%S`
export jobID
#WORK DIR
WORKDIR=${ROOT}/${jobID}
export WORKDIR
# save Variables Part 2 use
echo $confID > ${SPCTMP}/lastconfID
echo $jobID > ${SPCTMP}/lastjobID
echo $BMTPATH > ${SPCTMP}/BMTPATH
echo $BMTHOST > ${SPCTMP}/BMTHOST
echo $WORKDIR > ${SPCTMP}/WORKDIR
echo $JAVAPARMS > ${SPCTMP}/JAVAPARMS
echo $NSLAVES > ${SPCTMP}/NSLAVES
echo $EXPORTLOG > ${SPCTMP}/EXPORTLOG
echo $VDBENCH > ${SPCTMP}/VDBENCH
echo $JAVA > ${SPCTMP}/JAVA
echo $ROOT > ${SPCTMP}/ROOT
echo $SCRIPTS > ${SPCTMP}/SCRIPTS
echo $PREFILL > ${SPCTMP}/PREFILL
echo $BMTPATH > ${SPCTMP}/BMTPATH
echo $BMTHOST > ${SPCTMP}/BMTHOST
#
# setup the unique execution context(directory) for SPC1 job
#
cd ${ROOT}
mkdir ${jobID}
cd ${jobID}
# also for the slave host
ssh -T slaveh2 <<DERE
cd ${ROOT}
mkdir ${ROOT}/${jobID}
DERE
echo -e FDR for DX200S3 "\n"  start time `date` "\n" >
${SPCTMP}/${jobID}_message.txt
echo -e  confID=$confID "\n" >> ${SPCTMP}/${jobID}_message.txt
echo -e  jobID=$jobID "\n" >> ${SPCTMP}/${jobID}_message.txt
${SCRIPTS}/sendstatus.sh "Starting FDR Job=${jobID}" ${jobID}_message.txt

#
# Save WG Information on both slave servers (H1 and H2)
#
${SCRIPTS}/sendstatus.sh "Starting GetWGInfo Job=${jobID}" ${jobID}_message.txt
${SCRIPTS}/GetWGInfo.sh  &> ${jobID}_WG_Info_SlaveH1.txt
scp ${SCRIPTS}/GetWGInfo.sh root@slaveh2:GetWGInfo.sh
ssh -t -t  root@slaveh2 ./GetWGInfo.sh &> ${jobID}_WG_Info_SlaveH2.txt
```

```
${SCRIPTS}/sendstatus.sh "Completed GetWGInfo Job=${jobID}" ${jobID}_message.txt

#
# Create /dev/rawxxx mapping based on the LUNV subfield of VPD83 WWID
# i.e. /dev/raw1xx  - LUNV xx from port CA#0
# i.e. /dev/raw2xx  - LUNV xx from port CA#1
# copy the Mapping scripts to slave
scp ${SCRIPTS}/RawMapping.sh root@slaveh2:.
scp ${SCRIPTS}/GetLuns.sh root@slaveh2:.
scp ${SCRIPTS}/CleanupRaw.sh root@slaveh2:.
scp ${SCRIPTS}/DoRaw.awk root@slaveh2:.
cp ${SCRIPTS}/GetLuns.sh .
cp ${SCRIPTS}/CleanupRaw.sh .
cp ${SCRIPTS}/DoRaw.awk .
${SCRIPTS}/sendstatus.sh "Starting RawMapping Job=${jobID}" ${jobID}_message.txt
${SCRIPTS}/RawMapping.sh M1
ssh -t -t  root@slaveh2 ./RawMapping.sh S1
${SCRIPTS}/sendstatus.sh "Completed RawMapping Job=${jobID}" ${jobID}_message.txt
#
# Save beforeP log
#
${SCRIPTS}/sendstatus.sh "Starting BeforeP log save Job=${jobID}"
${jobID}_message.txt
${SCRIPTS}/${EXPORTLOG} ${jobID}_beforeP
cp /tmp/*${jobID}_beforeP* .
${SCRIPTS}/sendstatus.sh "Completed BeforeP log save Job=${jobID}"
${jobID}_message.txt
#
# Start Sequence of SPC1 benchmark jobs
#
# Setup the Java environment variables for SPC1
#
CLASSPATH=/usr/local/spc/spc1;export CLASSPATH
LD_LIBRARY_PATH=/usr/local/spc/spc1;export LD_LIBRARY_PATH
#
# setup for Prefill task by copying the prefill parameters file
#
${SCRIPTS}/sendstatus.sh "Starting Prefill test for Job=${jobID}"
${jobID}_message.txt
mkdir Prefill
cd Prefill
cp ${SCRIPTS}/${confID}_prefill.txt .
${VDBENCH} -f ${confID}_prefill.txt
cd ..
#
${SCRIPTS}/sendstatus.sh "Completed Prefill test for Job=${jobID}"
${jobID}_message.txt
# Collect Logs after the prefill job
${SCRIPTS}/sendstatus.sh "Starting  BeforeF log save Job=${jobID}"
${jobID}_message.txt
${SCRIPTS}/${EXPORTLOG} ${jobID}_BeforeF
cp /tmp/*${jobID}_BeforeF* .
${SCRIPTS}/sendstatus.sh "Completed BeforeF log save Job=${jobID}"
${jobID}_message.txt
#
# Setup headers and trailers for lowlevel parameter files and the slave files
${SCRIPTS}/Setup2FDR.sh ${BSU}
#
${SCRIPTS}/sendstatus.sh "Starting Metrics/Sustain test step for Job=${jobID}"
${jobID}_message.txt
#metrics
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  sustain
${SCRIPTS}/start_slaves.sh sustain
```

```
${JAVA} $JAVAPARMS spc1 -w SPC1   -f sustain.txt -o sustain SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/sendstatus.sh "Completed Metrics/Sustain test step for Job=${jobID}"
${jobID}_message.txt
${SCRIPTS}/sendstatus.sh "Starting Metrics/Ramp test step for Job=${jobID}"
${jobID}_message.txt
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  ramp100
${SCRIPTS}/start_slaves.sh ramp100
${JAVA} $JAVAPARMS spc1 -w SPC1  -f ramp100.txt -o ramp100 SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  ramp095
${SCRIPTS}/start_slaves.sh ramp095
${JAVA} $JAVAPARMS spc1 -w SPC1  -f ramp095.txt -o ramp095 SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  ramp090
${SCRIPTS}/start_slaves.sh ramp090
${JAVA} $JAVAPARMS spc1 -w SPC1  -f ramp090.txt -o ramp090 SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  ramp080
${SCRIPTS}/start_slaves.sh ramp080
${JAVA} $JAVAPARMS spc1 -w SPC1  -f ramp080.txt -o ramp080 SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  ramp050
${SCRIPTS}/start_slaves.sh ramp050
${JAVA} $JAVAPARMS spc1 -w SPC1  -f ramp050.txt -o ramp050 SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  ramp010
${SCRIPTS}/start_slaves.sh ramp010
${JAVA} $JAVAPARMS spc1 -w SPC1  -f ramp010.txt -o ramp010 SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/sendstatus.sh "Completed Metrics/Ramp test step for Job=${jobID}"
${jobID}_message.txt
#repeat-1
${SCRIPTS}/sendstatus.sh "Starting Repeatablity test 1 step for Job=${jobID}"
${jobID}_message.txt
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  repeat1_lrt
${SCRIPTS}/start_slaves.sh repeat1_lrt
${JAVA} $JAVAPARMS spc1 -w SPC1 -f repeat1_lrt.txt -o repeat1_lrt SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  repeat1_iops
${SCRIPTS}/start_slaves.sh repeat1_iops
${JAVA} $JAVAPARMS spc1 -w SPC1 -f repeat1_iops.txt -o repeat1_iops SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/sendstatus.sh "Completed Repeatablity test 1 step for Job=${jobID}"
${jobID}_message.txt
#repeat-2
${SCRIPTS}/sendstatus.sh "Starting Repeatablity test 2 step for Job=${jobID}"
${jobID}_message.txt
${SCRIPTS}/setup_slave_dirs.sh ${jobID}  repeat2_lrt
${SCRIPTS}/start_slaves.sh repeat2_lrt
${JAVA} $JAVAPARMS spc1 -w SPC1 -f repeat2_lrt.txt -o repeat2_lrt SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/setup_slave_dirs.sh ${jobID}   repeat2_iops
${SCRIPTS}/start_slaves.sh repeat2_iops
${JAVA} $JAVAPARMS spc1 -w SPC1 -f repeat2_iops.txt -o repeat2_iops SPCOut
${SCRIPTS}/stop_slaves.sh
${SCRIPTS}/sendstatus.sh "Completed Repeatablity test 2 step for Job=${jobID}"
${jobID}_message.txt
#persist-1 reduced BSU value $BSUPERSIST
${SCRIPTS}/sendstatus.sh "Starting Persistence test 1 step for Job=${jobID}"
${jobID}_message.txt
cp ${SCRIPTS}/spc1.cfg .
${JAVA} $JAVAPARMS persist1 -b  $BSUPERSIST
```

```
${SCRIPTS}/sendstatus.sh "Completed Persistence test 1 step for Job=${jobID}"
${jobID}_message.txt
# Setup the Java environment variables for SPC2
#
#
SPC2=/usr/local/spc/spc2
CLASSPATH=${SPC2};export CLASSPATH
LD_LIBRARY_PATH=${SPC2};export LD_LIBRARY_PATH
#
# Start sequence of benchmark jobs
#
#copy the parameter files to work directory
cp ${SCRIPTS}/parm_FDR*.txt .
#
#Pers-Write job
#
${SCRIPTS}/sendstatus.sh "Starting SPC2 Pers-W step for Job=${jobID}"
${jobID}_message.txt
${JAVA} $JAVAPARMS vdbench -w SPC2 -f parm_FDR_pers_write.txt -o out_Pers-Write -
init
${JAVA} $JAVAPARMS vdbench -w SPC2 -f parm_FDR_pers_write.txt -o out_Pers-Write
${SCRIPTS}/sendstatus.sh "Completed SPC2 Pers-W step for Job=${jobID}"
${jobID}_message.txt
# Perform Power cycle
${SCRIPTS}/sendstatus.sh "Array is Ready for manual Power cycle Job=${jobID} "
${jobID}_message.txt
```

doFDRall_2M.sh

```
#!/bin/bash -x
#
# Do 2nd steps required for the FDR run
#
# obtain jobID and confID saved from part 1
#
# check to see if previous context exists
# create tmp directory for spc1 if it does not exist
SPCTMP=/tmp/spc1
if [ ! -d ${SPCTMP} ]; then
 echo Error!
 exit
else
confID=`cat ${SPCTMP}/lastconfID`
jobID=`cat ${SPCTMP}/lastjobID`
BMTPATH=`cat ${SPCTMP}/BMTPATH`
BMTHOST=`cat ${SPCTMP}/BMTHOST`
WORKDIR=`cat ${SPCTMP}/WORKDIR`
JAVAPARMS=`cat ${SPCTMP}/JAVAPARMS`
NSLAVES=`cat ${SPCTMP}/NSLAVES`
EXPORTLOG=`cat ${SPCTMP}/EXPORTLOG`
VDBENCH=`cat ${SPCTMP}/VDBENCH`
JAVA=`cat ${SPCTMP}/JAVA`
ROOT=`cat ${SPCTMP}/ROOT`
SCRIPTS=`cat ${SPCTMP}/SCRIPTS`
PREFILL=`cat ${SPCTMP}/PREFILL`
fi
export SPCTMP
export confID
export jobID
export BMTPATH
export BMTHOST
export WORKDIR
export JAVAPARMS
export NSLAVES
export EXPORTLOG
export VDBENCH
export JAVA
export ROOT
export SCRIPTS
export PREFILL
#
cd ${WORKDIR}
#
# Create /dev/rawxxx mapping based on the LUNV subfield of VPD83 WWID
# i.e. /dev/raw1xx  - LUNV xx from port CA#0
# i.e. /dev/raw2xx  - LUNV xx from port CA#1
${SCRIPTS}/sendstatus.sh "Starting P2 RawMapping Job=${jobID}" ${jobID}_message.txt
${SCRIPTS}/RawMapping.sh M2
ssh -t -t  root@slaveh2 ./RawMapping.sh S2
${SCRIPTS}/sendstatus.sh "Completed P2 RawMapping Job=${jobID}" ${jobID}_message.txt
#
# Run persistence 2
#
CLASSPATH=/usr/local/spc/spc2;export CLASSPATH
LD_LIBRARY_PATH=/usr/local/spc/spc2;export LD_LIBRARY_PATH
${SCRIPTS}/sendstatus.sh "Starting SPC2 Persistence2 step for Job=${jobID}"
${jobID}_message.txt
#persist-2
${JAVA} $JAVAPARMS vdbench -w SPC2 -f parm_FDR_pers_read.txt -o out_Pers-Read
${SCRIPTS}/sendstatus.sh "Completed SPC2 Persistence2 step for Job=${jobID}"
${jobID}_message.txt
#
# Save after log
#
${SCRIPTS}/sendstatus.sh "Starting AfterJ log save Job=${jobID}"
${jobID}_message.txt
${SCRIPTS}/exportLog.exp ${jobID}_AfterJ
cp /tmp/*${jobID}_AfterJ* .
${SCRIPTS}/sendstatus.sh "Completed AfterJ log save Job=${jobID}"
${jobID}_message.txt
# Collect Archive
cd ${ROOT}
mv ${jobID} ${jobID}_Results
mkdir ${jobID}
sed s/##jobID##/${jobID}/  < README_Template.txt |sed s/##confID##/${confID}/ >
${jobID}/README_${jobID}.txt
tar -cf ${jobID}.tar [1-9]_*
mv ${jobID}.tar ${jobID}
cd ${jobID}
tar -xf ${jobID}.tar
rm -rf ${jobID}.tar
mv ../${jobID}_Results/* 9_Results/
mv 9_Results/log_*.zlg 4_Repository
mv 9_Results/cfg_*.cfg 4_Repository
mv 9_Results/*WG_Info*.txt  6_Tunables
mv 9_Results/*.txt  7_Execution
rm -rf ../${jobID}_Results
# Get slave outputs from the slave servers and merge them into the archive
ssh -T slaveh2 <<DERE
cd ${ROOT}
cd ${jobID}
tar -cf /tmp/${jobID}_slaves.tar .
cd /root
tar -cf /tmp/${jobID}_mapping.tar  hbalist_S[12] MakeRaw_S[12].sh
DERE
```

```
scp slaveh2:/tmp/${jobID}_slaves.tar /tmp
cd 9_Results
tar -x -k -f /tmp/${jobID}_slaves.tar .
scp slaveh2:/tmp/${jobID}_mapping.tar /tmp
tar -x -k -f /tmp/${jobID}_mapping.tar .
# save nohup and zip up the archive (exclude active file: nohup.out )
sync
cp ${SCRIPTS}/nohup.out nohup_${jobID}.out
cd ${ROOT}
zip -r ${jobID} ${jobID}/ -x ${jobID}/7_Execution/nohup.out
# Send Archive to repository
date=`date +%Y%m%d`
ssh -T $BMTHOST <<HERE
cd $BMTPATH
mkdir $date
HERE
scp ${jobID}.zip ${BMTHOST}:${BMTPATH}${date}/
${SCRIPTS}/sendstatus.sh "Completed FDR job All data in ${BMTHOST}
${BMTPATH}${date}/${jobID}.zip" ${jobID}_message.txt
# rm the global variables
rm -f ${SPCTMP}/lastconfID
rm -f ${SPCTMP}/lastjobID
rm -f ${SPCTMP}/BMTPATH
rm -f ${SPCTMP}/BMTHOST
rm -f ${SPCTMP}/WORKDIR
rm -f ${SPCTMP}/JAVAPARMS
rm -f ${SPCTMP}/NSLAVES
rm -f ${SPCTMP}/EXPORTLOG
rm -f ${SPCTMP}/VDBENCH
rm -f ${SPCTMP}/JAVA
rm -f ${SPCTMP}/ROOT
rm -f ${SPCTMP}/SCRIPTS
rm -f ${SPCTMP}/PREFILL
```

Referenced Scripts

sendstatus.sh

This script sends email to report status of the various benchmark execution steps.

```
#!/bin/bash
# send status <message> <file> <file2>
# send status message to the group
message=$1
file=$2
echo current time = `date` ":" >> /tmp/spc1/$file
echo -e "status = $message \n" >> /tmp/spc1/$file
if [ $# = 3 ]; then
file2=$3
cat ${file2} /tmp/spc1/${file} > /tmp/spc1/${file2}_T
scp /tmp/spc1/${file2}_T fjuser@129.212.198.24:spcbin/${file}
else
scp /tmp/spc1/$file fjuser@129.212.198.24:spcbin/$file
fi
ssh fjuser@129.212.198.24 spcbin/spcstatus.sh "\"$message\"" spcbin/$file
```

GetWGInfo.sh

Get various Host System configuration information.

```
function prtheader()
{
 echo "-------------------------------------------------------------"
 echo "--" $1 "--"
 echo "-------------------------------------------------------------"
}
prtheader "CPU INFORMATION"
cat /proc/cpuinfo
prtheader "MEMORY INFORMATION"
cat /proc/meminfo
prtheader "OPERATING SYSTEM INFORMATION"
lsb_release -a
prtheader "JAVA RUNTIME ENVIRONMENT INFORMAION"
java -version
prtheader "HBA CONFIGURATION INFORMATION"
qaucli -i
prtheader "HBA DRIVER CONFIGURATION FILE"
cat /etc/modprobe.d/qla2xxx.conf
prtheader "SCSI TARGET DEVICE INFORMATION"
for i in `qaucli -i |grep "Port Name" |awk -F ":" '{print $2}'`; do  echo
HBA_$i;qaucli -l $i; done
```

RawMapping.sh

This configuration uses Linux "raw" mechanism to bind a storage block device to a raw character device in order insure direct device access for the benchmark. Raw devices (/dev/dev/rawxxx) are mapped to Linux block devices (/dev/sd<x>). That mapping is not persistent between power cycles.

This set of scripts (**RawMapping.sh, GetLuns.sh, CleanupRaw.sh** and **DoRaw.awk**), invoked after every power cycle, ensures the correct relationship is maintained even if the mapping relationship between the array and Linux storage name changes.

```
#!/bin/bash
# RawMapping <Suffix>
# RawMapping the configuration and device files
#RawMapping.sh
# calls GetLuns.sh - Generate HBA information file
#       CleanupRaw.sh - Remove existing /dev/raw/ links
#       DoRaw.awk - Geneate a shell file to make raw links
if [ $# -ne 1 ]
then
 echo "$0 <Suffix>"
 exit
else
 Suffix=$1
fi
./GetLuns.sh $Suffix
awk -f DoRaw.awk /tmp/.lunlist > /tmp/.MakeRaw_${Suffix}.sh
sort /tmp/.MakeRaw_${Suffix}.sh > MakeRaw_${Suffix}.sh
chmod 777 MakeRaw_${Suffix}.sh
./CleanupRaw.sh
./MakeRaw_${Suffix}.sh
exit
```

GetLuns.sh

This script calls the ***qaucli*** command generate mapping information as seen by each initiator port of the HBA, which is saved in an output file, **hbslist<suffix>/HBA_<Initiator WWWPN>**.

```
#!/bin/bash
# GetLuns <suffix>
rm -rf hbalist_$1 &> /dev/null
mkdir hbalist_$1
cd hbalist_$1
for i in `qaucli -i |grep "Port Name" |awk -F ":" '{print $2}'`; do  echo
HBA_$i;qaucli -l $i > HBA_$i ; done
cd ..
rm -rf /tmp/.lunlist &> /dev/null
cat hbalist_$1/HBA_* > /tmp/.lunlist
```

CleanupRaw.sh

This script will issue the ***raw*** command a major and minor device number of zero (0) to clear up any existing bindings.

```
for i in `raw -qa|sed s/:.*$//` ;do raw $i 0 0; done
```

DoRaw.awk

This script will create the **MakeRaw <suffix>.sh** script, which generates the Linux raw command that binds the raw device with the Linux storage device. Once the **MakeRaw <suffix>.sh** script is created, it is executed.

```
#awk script to parse output of qlogic lunlist
BEGIN {FS = " "; RS="\n"}
/HBA Instance/&&RS!="Product Vendor"{
  iwwn=$8
#  print "IWWN="iwwn"\n";
}
/Target WWPN/&&RS!="Product Vendor"{
  twwn=$3
  FS = "\n"; RS="Product Vendor"
}
RS=="Product Vendor"{
 if ($1~/FUJITSU/)
 {
  split($5,siz,"    ");        #record    5    contains    size    in    GiB        Size
: 433.13 GB
  size=siz[3] * 0.60 * 1.07  # Space Utilization of 60% and 1.07 to adjust for GiB
vs. GB
  split($7,wwx,"-");  #record 7contains the VPD83 ID
  lunv= strtonum("0x" wwx[13] wwx[14]);
  split(twwn,twwx,"-"); #last hex digit of Target WWN is 5 then 200
  if(twwx[8]~/[54]/)
     lunv=lunv+200;
  else
     lunv=lunv+100;
  split($8,devn," ");split(devn[5],ddn,";");
  printf("raw /dev/raw/raw%s %s # Size=%d\n",lunv,ddn[1],size);
 }
 if($10~/Using/)
 {
  n=11
  while($n!~/HBA Instance/&&n<NF)
      n=n+1;
  if(n!=NF)
  {
      z=split($n,wwn," ");iwwn=wwn[8]

  }
  while($n!~/Target WWPN/&&n<NF)
      n=n+1;
  if(n!=NF)
  {
    z=split($n,wwn," ");twwn=wwn[3]
  }
 }
}
```

MakeRaw_M1.sh

This script is generated by do the DoRaw.awk script, described above, to call the **raw** utility to bind the raw device with the Linux block device. **M1** is a suffix designating the appropriate Host System.

```
  raw /dev/raw/raw100 /dev/sdb # Size=369
  raw /dev/raw/raw101 /dev/sdg # Size=369
  raw /dev/raw/raw102 /dev/sde # Size=369
  raw /dev/raw/raw103 /dev/sdk # Size=369
  raw /dev/raw/raw104 /dev/sdi # Size=369
  raw /dev/raw/raw105 /dev/sdo # Size=369
  raw /dev/raw/raw106 /dev/sdm # Size=369
  raw /dev/raw/raw107 /dev/sdr # Size=369
  raw /dev/raw/raw108 /dev/sds # Size=82
  raw /dev/raw/raw109 /dev/sdw # Size=82
  raw /dev/raw/raw110 /dev/sdx # Size=82
  raw /dev/raw/raw111 /dev/sdy # Size=82
  raw /dev/raw/raw200 /dev/sdc # Size=369
  raw /dev/raw/raw201 /dev/sdd # Size=369
  raw /dev/raw/raw202 /dev/sdf # Size=369
  raw /dev/raw/raw203 /dev/sdh # Size=369
  raw /dev/raw/raw204 /dev/sdj # Size=369
```

```
raw /dev/raw/raw205 /dev/sdl # Size=369
raw /dev/raw/raw206 /dev/sdn # Size=369
raw /dev/raw/raw207 /dev/sdp # Size=369
raw /dev/raw/raw208 /dev/sdq # Size=82
raw /dev/raw/raw209 /dev/sdt # Size=82
raw /dev/raw/raw210 /dev/sdu # Size=82
raw /dev/raw/raw211 /dev/sdv # Size=82
```

MakeRaw_M2.sh

This script is generated by do the **DoRaw.awk** script, described above, to call the **raw** utility to bind the raw device with the Linux block device. **M2** is a suffix designating the appropriate Host System.

```
raw /dev/raw/raw100 /dev/sdb # Size=369
raw /dev/raw/raw101 /dev/sdd # Size=369
raw /dev/raw/raw102 /dev/sdf # Size=369
raw /dev/raw/raw103 /dev/sdh # Size=369
raw /dev/raw/raw104 /dev/sdj # Size=369
raw /dev/raw/raw105 /dev/sdl # Size=369
raw /dev/raw/raw106 /dev/sdn # Size=369
raw /dev/raw/raw107 /dev/sdo # Size=369
raw /dev/raw/raw108 /dev/sdq # Size=82
raw /dev/raw/raw109 /dev/sdr # Size=82
raw /dev/raw/raw110 /dev/sdv # Size=82
raw /dev/raw/raw111 /dev/sdu # Size=82
raw /dev/raw/raw200 /dev/sdc # Size=369
raw /dev/raw/raw201 /dev/sde # Size=369
raw /dev/raw/raw202 /dev/sdg # Size=369
raw /dev/raw/raw203 /dev/sdi # Size=369
raw /dev/raw/raw204 /dev/sdk # Size=369
raw /dev/raw/raw205 /dev/sdm # Size=369
raw /dev/raw/raw206 /dev/sds # Size=369
raw /dev/raw/raw207 /dev/sdp # Size=369
raw /dev/raw/raw208 /dev/sdw # Size=82
raw /dev/raw/raw209 /dev/sdt # Size=82
raw /dev/raw/raw210 /dev/sdy # Size=82
raw /dev/raw/raw211 /dev/sdx # Size=82
```

Setup2FDR.sh

This script creates the SPC-1 configuration file used by the SPC-1 Persistence Test (**spc1.cfg**), the various Slave JVM configuration files and the various low-level SPC-1 Test Run configuration files.

The manually created input files required by the script are listed in the "**input files**" portion of the Referenced Files section below. The resulting output files, which have not been previously listed, also appear in the "**output files**" portion of the Referenced Files section below.

```
#!/bin/bash
#
# Setup2FDR.sh  <BSU>
# Setup the parameter files for FDR run
# Called by doFDRall_1M.sh
# Parameter  BSU - BSU used for FDR run
# Input files (generated manually)
#    spc1-device.cfg
#    sustain_trailer.txt
#    ramp100_trailer.txt
#    ramp095_trailer.txt
#    ramp090_trailer.txt
#    ramp080_trailer.txt
#    ramp050_trailer.txt
#    ramp010_trailer.txt
#    repeat1_lrt_trailer.txt
#    repeat1_iops_trailer.txt
#    repeat2_lrt_trailer.txt
#    repeat2_iops_trailer.txt
#
# Output files:
#    slave_header.txt
#    spc1_master_header.txt
#    sustain.txt
#    ramp100.txt
#    ramp095.txt
#    ramp090.txt
#    ramp080.txt
#    ramp050.txt
#    ramp010.txt
#    repeat1_lrt.txt
#    repeat1_iops.txt
#    repeat2_lrt.txt
#    repeat2_iops.txt
#
# Enviroment Variables
#  SCRIPTS
#  JAVAPARMS
#  NSLAVES
#
if [ $# != 1 ]
 then
   echo "$0 <Bsu>"
   exit
fi
MASTERIP=masterh
BSU=$1
cd $SCRIPTS
#Create header for slave configuration file
cat >slave_header.txt <<HERE
master=$MASTERIP
host=#slave1#
javaparms="$JAVAPARMS"
HERE
SLAVELIST=""
#Create header for master configuration file
for (( i=1; i<=NSLAVES; i++ ))
do
SLAVELIST=${SLAVELIST}slave${i}
if [ $i != $NSLAVES ]
 then
   SLAVELIST="${SLAVELIST},"
fi
done
cat > spc1_master_header.txt <<HERE
host=master
slaves=($SLAVELIST)
javaparms="$JAVAPARMS"
HERE
cat - spc1-device.cfg > spc1.cfg <<HERE
javaparms="$JAVAPARMS"
```

```
HERE
# Merge to generate the lowlevel master config files
BSU_095=$(( $BSU*19/20 ))
BSU_090=$(( $BSU*9/10 ))
BSU_080=$(( $BSU*8/10 ))
BSU_050=$(( $BSU/2 ))
BSU_010=$(( $BSU/10 ))
cat spc1_master_header.txt spc1-device.cfg sustain_trailer.txt |sed s/##BSU##/$BSU/
> sustain.txt
cat spc1_master_header.txt spc1-device.cfg ramp100_trailer.txt |sed s/##BSU##/$BSU/
> ramp100.txt
cat spc1_master_header.txt spc1-device.cfg ramp095_trailer.txt |sed
s/##BSU_095##/$BSU_095/ > ramp095.txt
cat spc1_master_header.txt spc1-device.cfg ramp090_trailer.txt |sed
s/##BSU_090##/$BSU_090/ > ramp090.txt
cat spc1_master_header.txt spc1-device.cfg ramp080_trailer.txt |sed
s/##BSU_080##/$BSU_080/ > ramp080.txt
cat spc1_master_header.txt spc1-device.cfg ramp050_trailer.txt |sed
s/##BSU_050##/$BSU_050/ > ramp050.txt
cat spc1_master_header.txt spc1-device.cfg ramp010_trailer.txt |sed
s/##BSU_010##/$BSU_010/ > ramp010.txt
cat spc1_master_header.txt spc1-device.cfg repeat1_lrt_trailer.txt |sed
s/##BSU_010##/$BSU_010/ > repeat1_lrt.txt
cat spc1_master_header.txt spc1-device.cfg repeat1_iops_trailer.txt |sed
s/##BSU##/$BSU/ > repeat1_iops.txt
cat spc1_master_header.txt spc1-device.cfg repeat2_lrt_trailer.txt |sed
s/##BSU_010##/$BSU_010/ > repeat2_lrt.txt
cat spc1_master_header.txt spc1-device.cfg repeat2_iops_trailer.txt |sed
s/##BSU##/$BSU/ > repeat2_iops.txt
```

setup_slave_dirs.sh

A separate execution directory is created for each low-level Test Run and each of the Slave JVMs used by that Test Run. The directories will contain the appropriate configuration file and workload generator output.

```
#!/bin/bash
# setup_slave_dirs <jobID> <STEP>
# Create the work directory and slave sub directory for each steop
# environment
#    ${ROOT}
#    ${WORKDIR}
#    ${SCRIPTS}
#    ${NSLAVES}
# parameter jobID STEP
jobID=$1
STEP=$2
cd ${WORKDIR}
mkdir ${STEP}
cd ${STEP}
# copy the config file for this step to the parent directory
cp ${SCRIPTS}/${STEP}.txt  ../${STEP}.txt
# create separate directory and  slave cfg files under each slave<n>/slave<n>.txt
for (( i=1; i<=NSLAVES ; i++ ))
do
 mkdir slave${i}
 cat ${SCRIPTS}/slave_header.txt ${SCRIPTS}/spc1-device.cfg  |sed
s/#slave1#/slave${i}/ > slave${i}/slave${i}.txt
done
#setup the directries in 2nd host
```

```
for (( i=$(($NSLAVES/2+1)); i<=$NSLAVES; i++ ))
do
tar -rf slavedirsH2.tar slave${i}
rm -rf slave${i}
done
scp slavedirsH2.tar  root@slaveh2:${WORKDIR}/slavedirsH2.tar
ssh -T slaveh2 <<DERE
cd ${WORKDIR}
mkdir ${STEP} >/dev/null
cd ${STEP}
tar -xf ${WORKDIR}/slavedirsH2.tar
DERE
```

start_slaves.sh

This script starts all of the Slave JVMs as the first step in each low-level Test Run. Each Slave JVM is started in the appropriate directory created in the previous script.

```
#!/bin/bash
#echo NSLAVES=${NSLAVES}
#echo PAVAPARMS=${JAVAPARMS}
#echo ROOT=${ROOT}
#echo jobID=${jobID}
STEP=$1
cd $WORKDIR
cd $STEP
#echo STEP=$STEP
for (( i=1; i<=$(($NSLAVES/2)); i++ ))
do
 (cd slave${i};java ${JAVAPARMS} spc1 -f slave${i}.txt &> slave${i}_console_out.txt)
&
 sleep 1
done
echo "#!/bin/bash" > start_slavesH2.sh
for (( i=$(($NSLAVES/2+1)); i<=$NSLAVES; i++ ))
do
 echo " ( cd slave${i};java ${JAVAPARMS} spc1 -f slave${i}.txt &>
slave${i}_console_out.txt ) & >/dev/null" >> start_slavesH2.sh
 echo "sleep 1" >> start_slavesH2.sh
done
chmod 777 start_slavesH2.sh
scp start_slavesH2.sh root@slaveh2:${ROOT}/${jobID}/${STEP}/start_slavesH2.sh
ssh -T  root@slaveh2 <<DHERE
cd ${ROOT}/${jobID}/${STEP}
./start_slavesH2.sh > /dev/null
DHERE
```

start_slavesH2.sh

This script is generated by the **start_slaves.sh** script to start the Slave JVMs on the second Host System.

```
#!/bin/bash
 ( cd slave26;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave26.txt &>
slave26_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave27;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave27.txt &>
slave27_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave28;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave28.txt &>
slave28_console_out.txt ) & >/dev/null
```

```
sleep 1
 ( cd slave29;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave29.txt &>
slave29_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave30;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave30.txt &>
slave30_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave31;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave31.txt &>
slave31_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave32;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave32.txt &>
slave32_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave33;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave33.txt &>
slave33_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave34;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave34.txt &>
slave34_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave35;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave35.txt &>
slave35_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave36;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave36.txt &>
slave36_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave37;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave37.txt &>
slave37_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave38;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave38.txt &>
slave38_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave39;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave39.txt &>
slave39_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave40;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave40.txt &>
slave40_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave41;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave41.txt &>
slave41_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave42;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave42.txt &>
slave42_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave43;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave43.txt &>
slave43_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave44;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave44.txt &>
slave44_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave45;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave45.txt &>
slave45_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave46;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave46.txt &>
slave46_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave47;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave47.txt &>
slave47_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave48;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave48.txt &>
slave48_console_out.txt ) & >/dev/null
sleep 1
 ( cd slave49;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave49.txt &>
slave49_console_out.txt ) & >/dev/null
```

```
sleep 1
 ( cd slave50;java -Xmx3072m -Xms3072m -Xss512k spc1 -f slave50.txt &>
slave50_console_out.txt ) & >/dev/null
sleep 1
```

stop_slaves.sh

The Slave JVMs are terminated at the end of each low-level Test Run.

```
ssh slaveh1 'pkill java'
ssh slaveh2 'pkill java'
```

Referenced Files

Input Files for Setup2FDR.sh

The following files were created manually as input: **spc1-device.cfg, sustain_trailer.txt, ramp100_trailer.txt, ramp095_trailer.txt, ramp090_trailer.txt, ramp080_trailer.txt, ramp050_trailer.txt, ramp010_trailer.txt, repeat1_lrt_trailer.txt, repeat1_iops_trailer.txt, repeat2_lrt_trailer.txt** and **repeat2_iops_trailer.txt**.

The content of each file appears below.

spc1-device.cfg

```
sd=asu1_1,lun=/dev/raw/raw100,size=369g #sdnum=1
sd=asu1_2,lun=/dev/raw/raw101,size=369g #sdnum=2
sd=asu1_3,lun=/dev/raw/raw203,size=369g #sdnum=3
sd=asu1_4,lun=/dev/raw/raw202,size=369g #sdnum=4
sd=asu2_1,lun=/dev/raw/raw104,size=369g #sdnum=5
sd=asu2_2,lun=/dev/raw/raw105,size=369g #sdnum=6
sd=asu2_3,lun=/dev/raw/raw207,size=369g #sdnum=7
sd=asu2_4,lun=/dev/raw/raw206,size=369g #sdnum=8
sd=asu3_1,lun=/dev/raw/raw109,size=82g #sdnum=9
sd=asu3_2,lun=/dev/raw/raw108,size=82g #sdnum=10
sd=asu3_3,lun=/dev/raw/raw210,size=82g #sdnum=11
sd=asu3_4,lun=/dev/raw/raw211,size=82g #sdnum=12
```

sustain_trailer.txt

```
rd=sustain,bsus=##BSU##,startup=180,elapsed=28800,interval=60
```

ramp100_trailer.txt

```
rd=ramp_100,bsus=##BSU##,startup=180,elapsed=600,interval=60
```

ramp095_trailer.txt

```
rd=ramp_95,bsus=##BSU_095##,startup=180,elapsed=600,interval=60
```

ramp090_trailer.txt

```
rd=ramp_90,bsus=##BSU_090##,startup=180,elapsed=600,interval=60
```

ramp080_trailer.txt

```
rd=ramp_80,bsus=##BSU_080##,startup=180,elapsed=600,interval=60
```

ramp050_trailer.txt

```
rd=ramp_50,bsus=##BSU_050##,startup=180,elapsed=600,interval=60
```

ramp010_trailer.txt

```
rd=ramp_10,bsus=##BSU_010##,startup=180,elapsed=600,interval=60
```

repeat1_lrt_trailer.txt

```
rd=repeat1_lrt,bsus=##BSU_010##,startup=180,elapsed=600,interval=60
```

repeat1_iops_trailer.txt

```
rd=repeat1_iops,bsus=##BSU##,startup=180,elapsed=600,interval=60
```

repeat2_lrt_trailer.txt

```
rd=repeat2_lrt,bsus=##BSU_010##,startup=180,elapsed=600,interval=60
```

repeat2_iops_trailer.txt

```
rd=repeat2_iops,bsus=##BSU##,startup=180,elapsed=600,interval=60
```

Output Files created by Setup2FDR.sh

The previously listed SPC-1 Workload configuration file (**spc1.cfg**) and two header files (**spc1_master_header.txt** and **slave_header.txt**), listed below, were created by **Setup2FDR.sh**.

The **spc1_master_header.txt** file was subsequently used by **Setup2FDR.sh** to create the previously listed SPC-1 Test Run configuration files. The **slave_header.txt** file was used by **setup_slave.dirs.sh** to create the previously listed Slave JVM configuration files.

spc1_master_header.txt

```
host=master
slaves=(slave1,slave2,slave3,slave4,slave5,slave6,slave7,slave8,slave9,slave10,slave
11,slave12,slave13,slave14,slave15,slave16,slave17,slave18,slave19,slave20,slave21,s
lave22,slave23,slave24,slave25,slave26,slave27,slave28,slave29,slave30,slave31,slave
32,slave33,slave34,slave35,slave36,slave37,slave38,slave39,slave40,slave41,slave42,s
lave43,slave44,slave45,slave46,slave47,slave48,slave49,slave50)
javaparms="-Xmx3072m -Xms3072m -Xss512k"
```

slave_header.txt

```
master=masterh
host=#slave1#
javaparms="-Xmx3072m -Xms3072m -Xss512k"
```