

INTERNATIONAL STANDARD

ISO/IEC
14776-341

First edition
2000-06

Information technology – Small Computer System Interface-3 (SCSI-3) – Part 341: Controller Commands (SCC)



Reference number
ISO/IEC 14776-341:2000(E)
Documents
AVA Document Sharing Hub

INTERNATIONAL STANDARD

ISO/IEC
14776-341

First edition
2000-06

Information technology – Small Computer System Interface-3 (SCSI-3) – Part 341: Controller Commands (SCC)

© ISO/IEC 2000

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland



PRICE CODE XB

Documents
For price, see our new catalogue
AVA Document Sharing Hub

PATENT STATEMENT

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this International Standard may involve the use of patents concerning SCC.

The ISO and IEC take no position concerning the evidence, validity and scope of this patent right. The holder of this putative patent right has assured the ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of the putative patent rights is registered with the ISO and IEC. Information may be obtained from:

IBM Corporation
Intellectual Property Licensing
5600 Cottle Road
San Jose, CA 95153

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Contents

	Page
Foreword	x
Introduction	xi
1 Scope	1
2 Normative references	1
2.1 Normative references	1
2.2 Approved references	1
2.3 References under development	1
3 Definitions, symbols, abbreviations, and conventions	1
3.1 Definitions	1
3.2 Abbreviations	4
3.3 Conventions	4
4 General	6
5 Models for systems containing arrays of devices	7
5.1 General	7
5.2 System layering model	7
5.2.1 SACL functions	7
5.2.2 Protocol conversion layer	8
5.2.3 Storage array conversion layer (SACL)	9
5.2.4 Examples of system layering variations	10
5.2.5 Branch of generic layers	11
5.2.6 Software SACL with a branch of SCSI disks	12
5.2.7 Branch with HBA SACL	13
5.2.8 Branch with bridge controller SACL	14
5.2.9 Branch with SACLs in multiple layers	15
5.3 Model for SCSI-3 storage arrays	16
5.3.1 SCSI storage array addressing	16
5.3.1.1 SCSI-3 storage array base address	16
5.3.1.2 Eight byte LUN structure	16
5.3.1.3 Component device address method	18
5.3.1.4 Logical unit address method	18
5.3.1.5 Peripheral device address method	19
5.3.1.6 Redundancy group address method	20
5.3.1.7 Spare address method	20
5.3.1.8 Volume set address method	20
5.3.2 SCSI-3 storage array objects	21
5.3.2.1 Adding objects	21
5.3.2.2 Association of objects	21
5.3.2.3 Attachment of objects	21
5.3.2.4 Covering of objects	22
5.3.2.5 Exchanging objects	22
5.3.2.6 Protected objects	22
5.3.2.7 Removing objects	22
5.3.2.8 Component device	23
5.3.2.9 Peripheral device	23
5.3.2.10 P_extent	23
5.3.2.11 Ps_extent	23
5.3.2.12 Redundancy group	24
5.3.2.12.1 No redundancy method of check data mapping	27
5.3.2.12.2 Copy redundancy method of check data mapping	27

5.3.2.12.3 XOR or P+Q redundancy method of check data mapping	27
5.3.2.12.4 Vendor specific redundancy method of check data mapping	27
5.3.2.13 Spares	27
5.3.2.14 Volume sets	28
5.3.3 SCSI-3 storage array operations	34
5.3.3.1 Rebuild operation	34
5.3.3.2 Recalculate operation	34
5.3.3.3 Regenerate operation	34
5.3.3.4 Verify operation	34
5.3.4 SCSI-3 storage array states	34
5.3.5 SCSI-3 storage array exception conditions	34
6 Commands for SCSI-3 storage array devices	36
6.1 Commands	36
6.2 MAINTENANCE(IN) command	37
6.2.1 MAINTENANCE(IN) command service actions	37
6.2.1.1 REPORT ASSIGNED/UNASSIGNED P_EXTENT service action	37
6.2.1.2 REPORT COMPONENT DEVICE service action	40
6.2.1.3 REPORT COMPONENT DEVICE ATTACHMENTS service	43
6.2.1.4 REPORT PERIPHERAL DEVICE service action	47
6.2.1.5 REPORT PERIPHERAL DEVICE ASSOCIATIONS service action	49
6.2.1.6 REPORT PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER service action	52
6.2.1.7 REPORT STATES service action	54
6.3 MAINTENANCE (OUT) commands	62
6.3.1 MAINTENANCE (OUT) command service actions	62
6.3.1.1 ADD PERIPHERAL DEVICE/COMPONENT DEVICE service action	63
6.3.1.2 ATTACH TO COMPONENT DEVICE service action	64
6.3.1.3 BREAK PERIPHERAL DEVICE/COMPONENT DEVICE service action	65
6.3.1.4 EXCHANGE P_EXTENT service action	66
6.3.1.5 EXCHANGE PERIPHERAL DEVICE/COMPONENT DEVICE service action	68
6.3.1.6 INSTRUCT COMPONENT DEVICE service action	69
6.3.1.7 REMOVE PERIPHERAL DEVICE/COMPONENT DEVICE service action	70
6.3.1.8 SET PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER service action	71
6.4 REDUNDANCY GROUP (IN) command	73
6.4.1 REDUNDANCY GROUP (IN) command service actions	73
6.4.1.1 REPORT REDUNDANCY GROUPS service action	73
6.4.1.2 REPORT UNASSIGNED REDUNDANCY GROUP SPACE service action	77
6.5 REDUNDANCY GROUP (OUT) command	80
6.5.1 REDUNDANCY GROUP (OUT) command service actions	80
6.5.1.1 CONTROL GENERATION OF CHECK DATA service action	81
6.5.1.2 CREATE/MODIFY REDUNDANCY GROUP service action	82
6.5.1.3 DELETE REDUNDANCY GROUP service action	85
6.5.1.4 REBUILD P_EXTENT service action	86
6.5.1.5 REBUILD PERIPHERAL DEVICE service action	88
6.5.1.6 RECALCULATE CHECK DATA service action	91
6.5.1.7 VERIFY CHECK DATA service action	92
6.6 VOLUME SET (IN) command	95
6.6.1 VOLUME SET (IN) command service actions	95
6.6.1.1 REPORT VOLUME SETS service action	95
6.7 VOLUME SET (OUT) command	98
6.7.1 VOLUME SET (OUT) command service actions	98
6.7.1.1 CONTROL GENERATION OF CHECK DATA service action	99
6.7.1.2 CONTROL WRITE OPERATIONS service action	100
6.7.1.3 CREATE/MODIFY VOLUME SET service action	101
6.7.1.4 DELETE VOLUME SET service action	104
6.7.1.5 RECALCULATE VOLUME SET CHECK DATA service action	105
6.7.1.6 VERIFY VOLUME SET CHECK DATA service action	107

6.8 SPARE (IN) command	111
6.8.1 SPARE (IN) command service actions	111
6.8.1.1 REPORT P_EXTENT SPARE service action	111
6.8.1.2 REPORT PERIPHERAL DEVICE/COMPONENT DEVICE SPARE service action	114
6.9 SPARE (OUT) command	117
6.9.1 SPARE (OUT) command service actions	117
6.9.1.1 CREATE/MODIFY P_EXTENT SPARE service action	118
6.9.1.2 CREATE/MODIFY PERIPHERAL DEVICE/COMPONENT DEVICE SPARE service action	121
6.9.1.3 DELETE SPARE service action	122
6.10 Parameters for direct-access devices	124
6.10.1 Mode parameters	124
6.10.1.1 LUN mapping page	124
Annex A SCSI-3 storage array IDENTIFY message format	126
A.1 SCSI-3 storage array IDENTIFY message overview	126
A.2 IDENTIFY message	126
Annex B SCSI-3 storage array addressing examples	127
B.1 SCSI storage array addressing examples overview	127
B.2 Addressing Examples for the 8-byte LUN structure	127
B.3 Addressing Examples for the 6-bit LUN structure	130
Annex C Examples of check data and user data mappings	132
C.1 Example XOR redundancy mapping	132
C.2 User data mapping examples	133

Tables

	Page
1 Addressing methods within a SCSI-3 storage array	16
2 Eight byte LUN structure adjustments	17
3 Eight byte LUN structure	17
4 FIRST LEVEL ADDRESSING field, SECOND LEVEL ADDRESSING field, THIRD LEVEL ADDRESSING field, and FOURTH LEVEL ADDRESSING field	18
5 ADDRESS METHOD	18
6 Logical unit addressing	19
7 Peripheral device addressing	19
8 Volume set addressing	20
9 Commands for SCSI storage array devices	36
10 Service actions for MAINTENANCE(IN) command	37
11 REPORT ASSIGNED/UNASSIGNED P_EXTENT service action	38
12 REPORT ASSIGNED/UNASSIGNED P_EXTENT parameter list	39
13 Data format of ASSIGNED/UNASSIGNED P_EXTENT DESCRIPTOR	39
14 Data format of P_EXTENT DESCRIPTOR	40
15 REPORT COMPONENT DEVICE service action	41
16 REPORT COMPONENT DEVICE parameter list	42
17 Data format of COMPONENT DEVICE DESCRIPTOR	42
18 COMPONENT DEVICE TYPES	43
19 REPORT COMPONENT DEVICE ATTACHMENTS service action	44
20 REPORT COMPONENT DEVICE ATTACHMENTS parameter list	45
21 Format of COMPONENT DEVICE ATTACHMENT DESCRIPTOR	46
22 Data format of LOGICAL UNIT DESCRIPTOR	46
23 LOGICAL UNIT types	47
24 REPORT PERIPHERAL DEVICE service action	47
25 SELECT REPORT	48
26 REPORT PERIPHERAL DEVICE parameter list	48
27 Format of PERIPHERAL DEVICE DESCRIPTOR	49
28 REPORT PERIPHERAL DEVICE ASSOCIATIONS service action	50
29 REPORT PERIPHERAL DEVICE ASSOCIATIONS parameter list	51
30 Format of PERIPHERAL DEVICE ASSOCIATIONS DESCRIPTOR	52
31 REPORT PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER service action	53
32 REPORT PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER parameter list	54
33 REPORT STATES service action	55
34 REPORT STATES	55
35 REPORT STATES parameter list	56
36 Format of LOGICAL UNIT STATES DESCRIPTORS	57
37 Target base device (LUN 0) states	58
38 Volume set states	59
39 Redundancy group states	60
40 Peripheral device and p_extent states	61
41 Spare states	61
42 Component device states	62
43 Service actions for MAINTENANCE (OUT) command	62
44 ADD PERIPHERAL DEVICE/COMPONENT DEVICE service action	63
45 ATTACH COMPONENT DEVICE service actions	64
46 ATTACH COMPONENT DEVICE parameter list	65
47 BREAK PERIPHERAL DEVICE/COMPONENT DEVICE service action	66
48 EXCHANGE P_EXTENT service action	67
49 EXCHANGE P_EXTENT parameters list	67
50 EXCHANGE PERIPHERAL DEVICE/COMPONENT DEVICE Service Action	68
51 INSTRUCT COMPONENT DEVICE service action	69
52 COMPONENT DEVICE INSTRUCTION field	69
53 INSTRUCT COMPONENT DEVICE parameter list	70
54 REMOVE PERIPHERAL DEVICE/COMPONENT DEVICE service action	70

55 SET PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER service action	71
56 SET PERIPHERAL DEVICE/COMPONENT DEVICE IDENTIFIER parameter list	72
57 Service actions for REDUNDANCY GROUP (IN) command	73
58 REPORT REDUNDANCY GROUPS service action	73
59 REPORT REDUNDANCY GROUPS parameter list	74
60 Format of REPORT REDUNDANCY GROUP DESCRIPTOR	75
61 REDUNDANCY GROUP IDENTIFIERS	75
62 GRANULARITY OF UNITS	76
63 REDUNDANCY GROUP P_EXTENT DESCRIPTOR	76
64 REPORT UNASSIGNED REDUNDANCY GROUP SPACE service action	77
65 REPORT UNASSIGNED REDUNDANCY GROUP SPACE parameter list	78
66 Format of REPORT UNASSIGNED REDUNDANCY GROUP SPACE DESCRIPTOR	79
67 Data format of PS_EXTENT DESCRIPTOR	80
68 Service actions for REDUNDANCY GROUP (OUT) command	81
69 CONTROL GENERATION OF CHECK DATA service action	81
70 CREATE/MODIFY REDUNDANCY GROUP service action	82
71 CREATE/MODIFY REDUNDANCY GROUP parameter list	83
72 Data format of CREATE/MODIFY P_EXTENT DESCRIPTOR	84
73 DELETE REDUNDANCY GROUP service action	85
74 REBUILD P_EXTENT service action	86
75 Rebuild types	87
76 REBUILD P_EXTENT parameter list	88
77 REBUILD PERIPHERAL DEVICE service action	89
78 Rebuild types	90
79 REBUILD PERIPHERAL DEVICE parameter list	91
80 RECALCULATE CHECK DATA service action	92
81 VERIFY CHECK DATA service action	93
82 Service actions for volume set (in) command	95
83 REPORT VOLUME SETS service action	95
84 REPORT VOLUME SETS parameter list	96
85 Format of REPORT VOLUME SET DESCRIPTOR	97
86 VOLUME SET PS_EXTENT DESCRIPTOR	98
87 Service actions for VOLUME SET (OUT) command	99
88 CONTROL GENERATION OF CHECK DATA service action	99
89 CONTROL WRITE OPERATIONS service action	101
90 CREATE/MODIFY VOLUME SET service action	102
91 CREATE/MODIFY VOLUME SET parameter list	103
92 Data format of CREATE/MODIFY PS_EXTENT DESCRIPTOR	104
93 DELETE VOLUME SET service action	105
94 RECALCULATE VOLUME SET CHECK DATA service action	106
95 RECALCULATE VOLUME SET CHECK DATA parameter list	107
96 VERIFY VOLUME SET CHECK DATA service action	108
97 VERIFY RANGE	109
98 VERIFY VOLUME SET CHECK DATA parameter list	110
99 Service actions for SPARE (IN) command	111
100 REPORT P_EXTENT SPARE service action	111
101 REPORT P_EXTENT SPARE parameter list	112
102 Format of REPORT P_EXTENT SPARE DESCRIPTOR	113
103 REPORT PERIPHERAL DEVICE/COMPONENT DEVICE SPARE service action	114
104 REPORT PERIPHERAL DEVICE/COMPONENT DEVICE SPARE parameter list	115
105 Format of REPORT PERIPHERAL DEVICE/COMPONENT DEVICE SPARE DESCRIPTOR	116
106 Data format of covered LOGICAL UNIT DESCRIPTOR	117
107 LOGICAL UNIT TYPES	117
108 Service actions for spare (out) command	118
109 CREATE/MODIFY P_EXTENT SPARE service action	118
110 CREATE/MODIFY P_EXTENT SPARE parameter list	120
111 CREATE/MODIFY PERIPHERAL DEVICE/COMPONENT DEVICE SPARE service action	121

112 CREATE/MODIFY PERIPHERAL DEVICE/COMPONENT DEVICE SPARE parameter list 122

113 DELETE SPARE service action 123

114 Mode page codes 124

115 LUN mapping page 125

A.1 IDENTIFY message format 126

Document Sharing Hub

Figures

	Page
0 SCSI-3 document road map	xii
1 Protocol conversion layer	8
2 SACL conversion layer	9
3 Typical system diagram	10
4 Branch of generic layers	11
5 Software SACL with a branch of SCSI disks	12
6 Branch with HBA SACL	13
7 Branch with bridge controller SACL	14
8 Branch with SACLs in multiple layers	15
9 Single redundancy group	24
10 Multiple volume sets associated with a single redundancy group	25
11 Redundancy group check data mapping flow chart	26
12 Multiple redundancy groups	29
13 Single volume set associated with multiple redundancy groups	29
14 Volume set user data mapping flow chart (part 1)	31
15 Volume set user data mapping flow chart (part 2)	32
16 Volume set user data mapping flow	33
C.1 XOR redundancy mapping example	132
C.2 User data mapping for a RAID 5 configuration	133
C.3 User data mapping for a RAID 3 configuration	134

Foreword

ISO (the International Organization of Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of international standard through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 14776-341 was prepared by subcommittee 25: Interconnection of information technology equipment, of ISO/IEC joint technical committee 1: Information technology.

Introduction

The SCSI command set is designed to provide efficient peer-to-peer operation of input/output devices (disks, tapes, printers, etc.) by an operating system. The SCSI command set assumes a command-response protocol. Action on SCSI commands shall not be deemed completed until a response is received. The response shall include a status that indicates the final disposition of the command.

The SCSI command set provides multiple operating systems concurrent control over one or more input/output devices. However, the multiple operating systems must properly coordinate their actions or data corruption will result. This standard defines commands that assist with coordination between multiple operations systems. However, details of the coordination are beyond the scope of the SCSI command set.

This standard defines a device model for SCSI-3 storage arrays. This standard defines the SCSI commands that may apply to SCSI-3 storage arrays and the SCSI commands that are uniquely for SCSI-3 storage arrays.

The SCSI-3 Controller Commands (SCC) standard is divided into six clauses:

- Clause 1 is the scope;
- Clause 2 enumerates the normative references that apply to this standard;
- Clause 3 describes the definitions, symbols and abbreviations used in this standard;
- Clause 4 describes the conceptual relationship between this document and the SCSI-3 Architecture Model;
- Clause 5 describes the command model for SCSI-3 storage array devices;
- Clause 6 defines the commands that may be implemented by an SCSI-3 storage array device.

The annexes provide information to assist with implementation of the SCSI-3 Controller Commands standard. Annex A is normative and is considered part of this standard. Annexes B and C are for information only.

Figure 0 is intended to show the relationship of this document to other SCSI-3 standards. The figure is not intended to imply a relationship such as a hierarchy, protocol stack, or system architecture. It indicates the applicability of a standard to the implementation of a given transport.

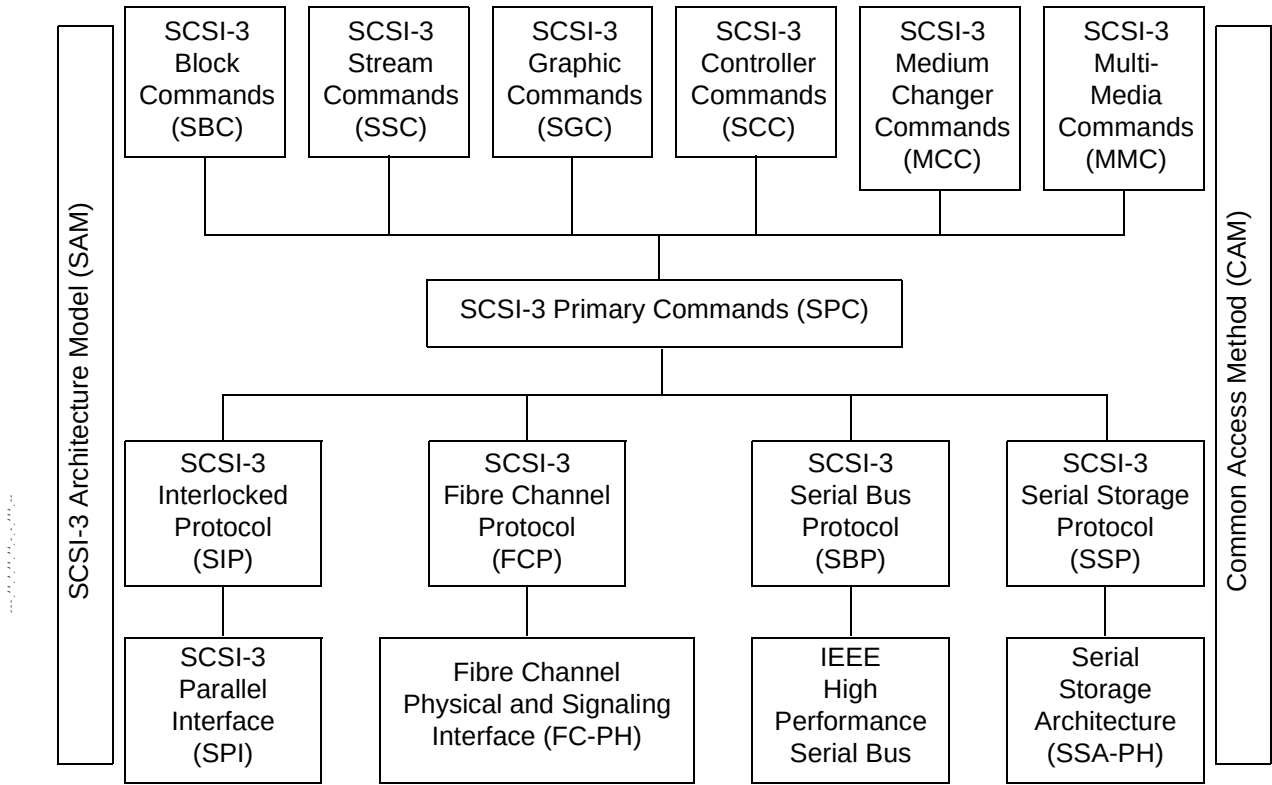


Figure 0 - SCSI-3 document road map

1 Scope

This International Standard defines the command set extensions for SCSI-3 storage array devices; commonly known as RAID devices. This standard is principally intended to be used in conjunction with, not as an alternate to, any of the SCSI-3 command standards and the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411). The resulting commands facilitate the control and configuration of SCSI-3 storage arrays and thus provide a common command specification for both system integrators and suppliers of SCSI-3 storage array devices.

The clause(s) of this standard pertaining to the SCSI-3 storage array device class, implemented in conjunction with the applicable clauses within any of the SCSI-3 command standards, shall specify the standard command set available for SCSI-3 storage arrays.

The objectives of the SCSI-3 Controller Commands is to provide the following:

- a) Transfer commands unique to SCSI-3 Controller Command devices;
- b) Control commands to manage the operation of an SCSI-3 Controller Command device;
- c) Optional device mapping and pass-through support.

2 Normative references

2.1 Normative references

The following standards contain provisions which, through reference in the text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below.

Members of IEC and ISO maintain registers of currently valid standards.

2.2 Approved references

SCSI-3 Architecture Model Standard (ISO/IEC 14776-411)

2.3 References under development

At the time of publication, the following referenced standards were still under development. For information on the current status of the document, or regarding availability, contact the relevant standards body or other organization as indicated.

SCSI-3 Interlocked Protocol Standard (ISO/IEC 14776-211)

SCSI-3 Primary Commands Standard (ISO/IEC 14776-311)

3 Definitions, symbols, abbreviations, and conventions

3.1 Definitions

For the purposes of this International standard the following definitions apply.

3.1.1 application client: An object that is the source of SCSI commands.

Note: Further definition of an application client can be found in the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411).

3.1.2 association: The linking of SCSI-3 storage array objects in a manner explicitly defined within this standard.

3.1.3 attachment: The linking of SCSI-3 storage array objects in a vendor specific manner.

3.1.4 base address: The LUN that an application client addresses to configure an SCSI-3 storage array and to determine information about an SCSI-3 storage array and the logical units attached to it.

3.1.5 byte: Indicates an 8-bit construct.

3.1.6 check data: Information contained within a redundancy group that allows lost or destroyed user data to be recreated.

3.1.7 check data mapping: The distribution of the check data within a redundancy group.

3.1.8 command: A request describing a unit of work to be performed by a device server.

Note: See the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) for a detailed definition of a command.

3.1.9 command descriptor block: The structure of up to 16 bytes used to communicate commands from an initiator to a target.

3.1.10 component device: Any physical addressable component not identifiable as an SCSI-3 peripheral device type.

Note: See table 18 for a list of components.

3.1.11 configuration: A collection of SCSI-3 storage array objects that follow the rules defined within this standard.

Note: For a list of SCSI-3 storage array objects see 5.3.2.

3.1.12 covering: The linking of spare objects to other SCSI-3 storage array objects in a manner explicitly defined within this standard.

3.1.13 field: A group of one or more contiguous bits.

3.1.14 initiator: An SCSI device containing application clients that originate device service requests to be processed by a target SCSI device.

Note: See the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) for a detailed definition of an initiator.

3.1.15 invalid: An illegal or unsupported field or code value.

3.1.16 groups: Objects that are independent from one another that may overlay one another.

3.1.17 logical block address: An address of a unit of data supplied or requested by an initiator.

3.1.18 logical unit: An externally addressable entity within a target that implements an SCSI device model.

Note: See the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) for a detailed definition of a logical unit.

3.1.19 logical unit identifier: An object that is part of the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) definition of a logical unit. A logical unit identifier uniquely identifies a logical unit in a SCSI domain.

Note: See the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) for a detailed definition of SCSI domain and logical unit identifier.

3.1.20 logical unit number: An identifier for a logical unit.

3.1.21 mandatory: The referenced item is required to claim compliance with this standard.

3.1.22 one: A true signal value or a true condition of a variable.

3.1.23 optional: The referenced item is not required to claim compliance with this standard. Implementation of an optional item shall be as defined in this standard.

3.1.24 page: Several commands use regular parameter structures that are referred to as pages. These pages are identified with a value known as a page code.

3.1.25 p_extent: All or part of the host addressable space within a single peripheral device of a SCSI-3 storage array.

3.1.26 peripheral device: Any addressable device identifiable as a SCSI-3 peripheral device type.

Note: See the IDENTIFY COMMAND description in the SCSI-3 Primary Commands Standard (ISO/IEC IS 14776-311) for the list of SCSI-3 peripheral device types.

3.1.27 protected space: The portion of a redundancy group that does not contain check data.

3.1.28 ps_extent: All or part of the protected space within a single peripheral device configured as a redundancy group.

3.1.29 rebuild operation: Recreation of protected space contents or any check data within a p_extent using check data and protected space contents from the remaining p_extents within the redundancy group.

3.1.30 recalculate operation: Recreation of check data from protected space contents.

3.1.31 redundancy group: A grouping of protected space and associated check data (Check data may be null) into a single logical unit that shall only have a single type of redundancy.

3.1.32 regenerate operation: Recreation of inaccessible protected space contents from accessible check data and protected space contents.

3.1.33 reserved: Identifies bits, fields, and code values that are set aside for future standardization.

3.1.34 SCSI-3 storage array: A target that processes SCSI command descriptor blocks and performs the services of a SACL. A single SCSI-3 storage array may contain multiple SACLs.

3.1.35 service action: A request describing a unit of work to be performed by a device server. A service action is an extension of a command.

Note: See the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) for a detailed definition of a command.

3.1.36 set: Objects that do not intersect and are independent from one another. Sets may span more than one device. A single device may contain more than one set or may contain the entire set.

3.1.37 spare: A range of logical block addresses, a component device, or a peripheral device covered by one or more redundancy groups, component devices, or peripheral devices that can be used to replace all or part of a redundancy group or a peripheral device or all of a component device.

3.1.38 storage array conversion layer (SACL): Converts input logical unit numbers to output logical unit numbers and may convert input logical block addresses to output logical block addresses.

3.1.39 stripe: All or part of a volume set that is bounded by a number of contiguous units within a single ps_extent and by a number of ps_extents.

3.1.40 target: In this standard a target refers to an SCSI-3 storage array device that performs an operation requested by an application client.

3.1.41 underlying redundancy group: The portion of a redundancy group that contains protected space that has been mapped to specific volume set(s).

3.1.42 unit: A standard basic quantity in bits, bytes, words, logical blocks, etc., specified by the GRANULARITY OF UNITS field (see table 62).

3.1.43 user data: The addressable logical blocks that are input to the SACL. Check data is not part of the addressable logical blocks.

3.1.44 user data mapping: The distribution of user data within a volume set.

3.1.45 vendor-specific: Something (e.g., a bit, field, code value, etc.) that is not defined by this standard and may be used differently in various implementations.

3.1.46 verify operation: Recreation of check data from protected space contents and the comparison of the recreated check data with the current check data.

3.1.47 volume set: One or more ps_extents grouped into a single LUN_V.

3.1.48 zero: A false signal value or a false condition of a variable.

3.2 Abbreviations

HBA	Host bus adapter
ITTU	I'm talking to you
LBA_P	Peripheral device logical block address
LBA_PS	Protected space logical block address
LBA_V	Volume set logical block address
LSB	Least significant bit
LUI	Logical unit identifier
LUN	Logical unit number
LUN_C	Component device logical unit number
LUN_P	Peripheral device logical unit number
LUN_R	Redundancy group logical unit number
LUN_S	Spare logical unit number
LUN_V	Volume set logical unit number
MSB	Most significant bit
SACL	Storage array conversion layer
SCSI	Either SCSI-2 or SCSI-3
SCSI-2	The Small Computer System Interface - 2 (ISO/IEC 9316)
SCSI-3	The Small Computer System Interface - 3
SIM	SCSI interface module

3.3 Conventions

Certain words and terms used in this International Standard have a specific meaning beyond the normal English meaning. These words and terms are defined either in 3.1 or in the text where they first appear. Names of messages, commands, statuses, sense keys, additional sense codes, and additional sense code qualifiers are in all uppercase (e.g., REQUEST SENSE), names of fields are in small uppercase (e.g., STATE OF SPARE), lower case is used for words having the normal English meaning.

Fields containing only one bit are usually referred to as the name bit instead of the name field.

Field names are in SMALL CAPS to distinguish them from normal English.

Numbers that are not immediately followed by lower-case b or h are decimal values.

Numbers immediately followed by lower-case b (xxb) are binary values.

Numbers immediately followed by lower-case h (xxh) are hexadecimal values

The term SCSI is used wherever it is not necessary to distinguish between the versions of SCSI. The

Small Computer System Interface - 2 (ISO/IEC 9316), is referred to herein as SCSI-2.

The term SCSI-3 refers collectively to the following documents:

- SCSI-3 Parallel Interface Standard (ISO/IEC IS 14776-111)
- SCSI-3 Interlocked Protocol Standard (ISO/IEC IS 14776-211)
- SCSI-3 Fiber Channel Protocol Standard (ISO/IEC IS 14776-221)
- SCSI-3 Serial Bus Protocol Standard (ISO/IEC IS 14776-231)
- SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411)
- SCSI-3 Primary Commands Standard (ISO/IEC IS 14776-311)
- SCSI-3 Block Commands Standard (ISO/IEC IS 14776-321)
- SCSI-3 Stream Commands Standard (ISO/IEC IS 14776-331)]
- SCSI-3 Graphic Commands Standard (ISO/IEC IS 14776-391)
- SCSI-3 Medium Changer Commands Standard (ISO/IEC IS 14776-351)
- SCSI-3 Multi-Media Commands Standard (ISO/IEC IS 14776-361)

4 General

This standard defines a device model for SCSI-3 storage arrays and the SCSI commands that apply to SCSI-3 storage arrays. This standard assumes all interconnects between devices are SCSI interconnects.

The SCSI command set assumes a command-response protocol. The fundamental properties of the command-response protocol are defined in the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411). In accordance with the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411), the command-response protocol can be modeled as a procedure call, specifically:

Service response = execute command (task identifier, command descriptor block, [data-out buffer], task attributes, || [data-in buffer], [autosense data], [autosense return flag], status).

The SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) defines all of the inputs and outputs in the procedure call. As they may apply to any SCSI device, this standard defines the contents of the following procedure call inputs and outputs; command descriptor block, data-out buffer, data-in buffer, and autosense data. This standard does not define all possible instances of these procedure inputs and outputs. This standard defines only those instances that may apply to SCSI-3 storage array devices.

This standard references values returned via the status output parameter. Examples of such status values are CHECK CONDITION and COMMAND TERMINATED. Status values are not defined by this standard, The SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411) defines all status values.

The entity that makes the procedure call from an initiator is an application client, as defined by the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411). The procedure call's representation arrives at the target in the form of a device service request. The entity that performs the work of the procedure call in a target is a device server. A device server is an object within a logical unit and is defined by the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411).

5 Models for systems containing arrays of devices

5.1 General

The first part of this clause defines a system layering model that uses the concept of SACLS to control arrays of devices. The second part of this clause defines the model for SCSI-3 storage array devices. The model assumes all the SCSI peripheral devices controlled within a SCSI-3 storage array are either fixed block or variable block devices.

5.2 System layering model

5.2.1 SACL functions

A SACL initiates several functions when an application client requests a media access. The type of media access and the configuration in effect determine which functions are used. The following are the functions available within a SACL:

- a) Translation of input logical unit identifiers to output logical unit identifiers;
- b) Translation of input logical block addresses to output logical block addresses;
- c) Reading data from and writing data to locations based on the configuration in effect for the addressed volume set;
- d) Calculating and updating the check data (if any);
- e) Regeneration of protected space contents within the volume set using check data information or duplicate data;
- f) Rebuilding of protected space contents associated with the redundancy group and (or) check data within the redundancy group using the contents of the redundancy group;
- g) Recalculation of the check data within a redundancy group;
- h) Recalculation of the check data within any redundancy group underlying a volume set;
- i) Determining when a p_extent should be disabled and/or replaced;
- j) Returning a confirmation to the application client as to the success or failure of a request and, in the case of a failure, gives possible corrective actions.

5.2.2 Protocol conversion layer

A system is typically composed of many protocol conversion layers (see figure 1), and these layers may exist in hardware or software. Each of these layers has input(s) and output(s). The next layer accessed is determined by the preceding layer's output.

These protocol conversion layers include, but are not limited to: transport modules, host adapter drivers, SIMs (SCSI interface module), HBAs (host bus adapter), bridge controllers, and storage drives. In this model each of these layers will be represented by a simple block that has an input and output.

All requests to or from a SACL contain logical unit identifiers but not all requests contain logical block addresses.

NOTE 1 - The logical unit identifier is defined in the SCSI-3 Architecture Model Standard (ISO/IEC IS 14776-411). SIMs and HBAs are defined in the SCSI-2 Common Access Method Transport and SCSI Interface Module Standard (ISO/IEC IS 9316-2).

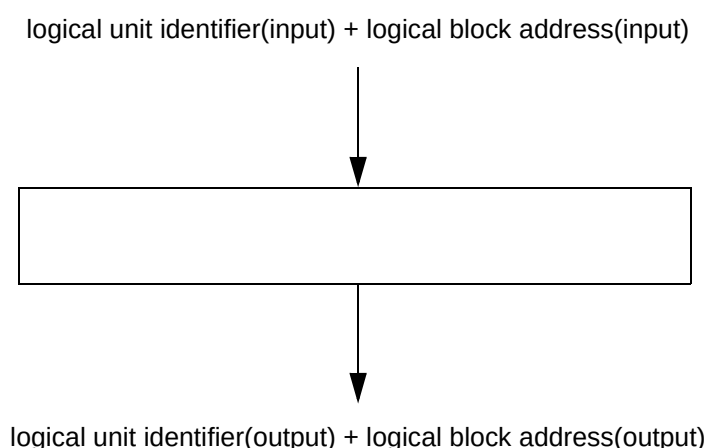


Figure 1 - Protocol conversion layer

Generic Layers do not modify the logical unit identifier or the logical block address.

NOTE 2 - There are types of layers other than generic and SACLs, however, these are not covered in this model.

5.2.3 Storage array conversion layer (SACL)

The SACL is capable of extensive manipulation on the logical unit identifier and the logical block address, based upon a consistent algorithm that follows the defined configuration. It is possible that a single logical unit identifier input or logical block address input may be converted to multiple different logical unit identifier outputs and logical block address outputs. In the following figures (figure 2, figure 3, and figure 5 through figure 8) SACLs will be shown with 'SACL' in the block.

The model does not require a one to one correspondence between logical unit identifiers and SCSI devices.

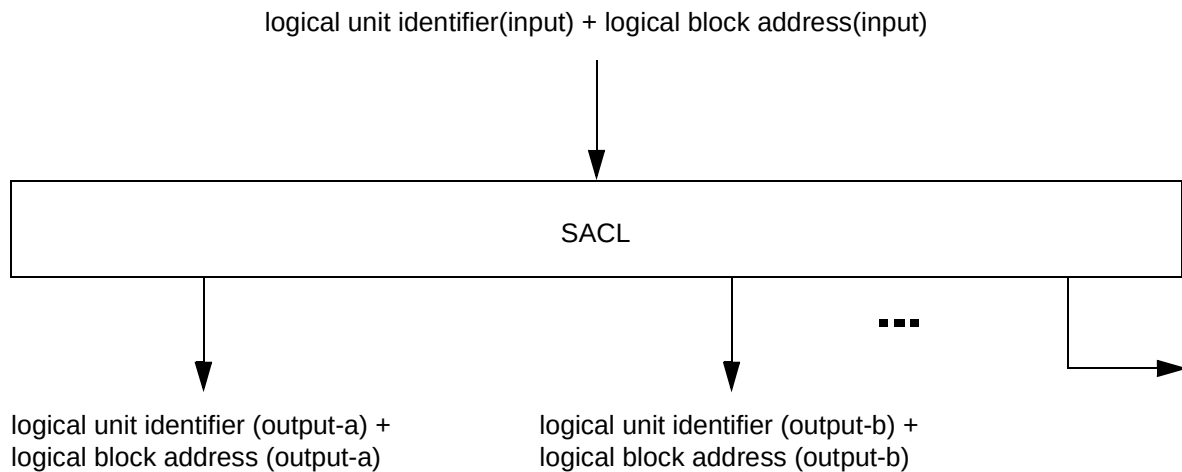


Figure 2 - SACL conversion layer

5.2.4 Examples of system layering variations

Typically a system diagram will be composed of many layers combined into a tree. For example, a driver may connect to multiple HBAs, which in turn may connect to multiple SCSI devices, etc. See figure 3 for an example of a system that consists of:

- a) one initiator that has two SCSI devices attached on a single SCSI bus that is not expandable;
- b) one initiator has two SCSI devices attached on a single SCSI that is expandable. One of the SCSI devices contains a SACL;
- c) the SCSI device that contains the SACL has three SCSI buses with SCSI devices attached and is capable of driving more SCSI buses:
 - a) two of the SCSI buses contain two SCSI devices each and these SCSI buses are not expandable;
 - b) one of the SCSI buses contains one SCSI device and is expandable.

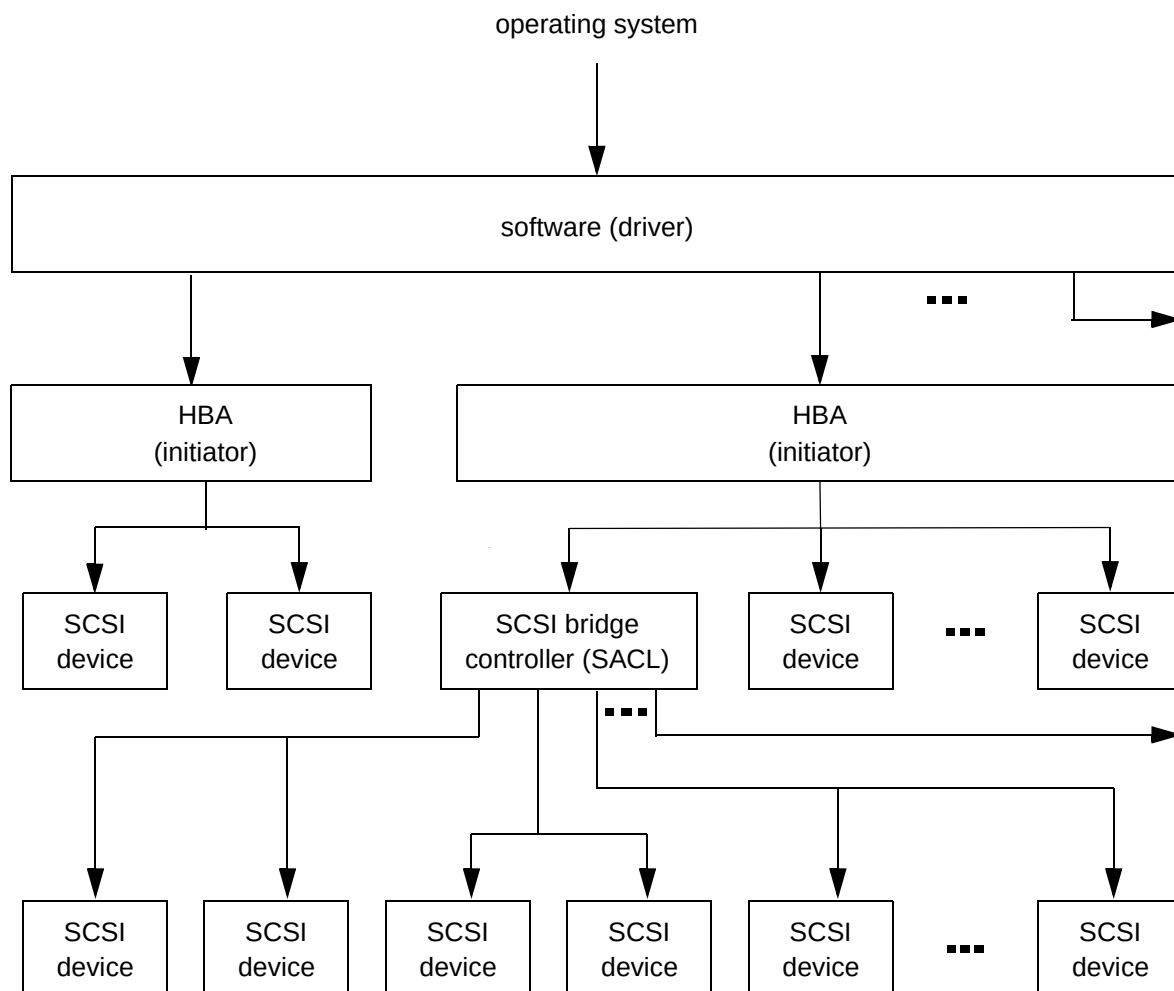


Figure 3 - Typical system diagram

5.2.5 Branch of generic layers

Figure 4 shows a system that does not contain any SCSI-3 storage arrays. In such a system all layers pass the logical unit identifier and logical block addresses directly through.

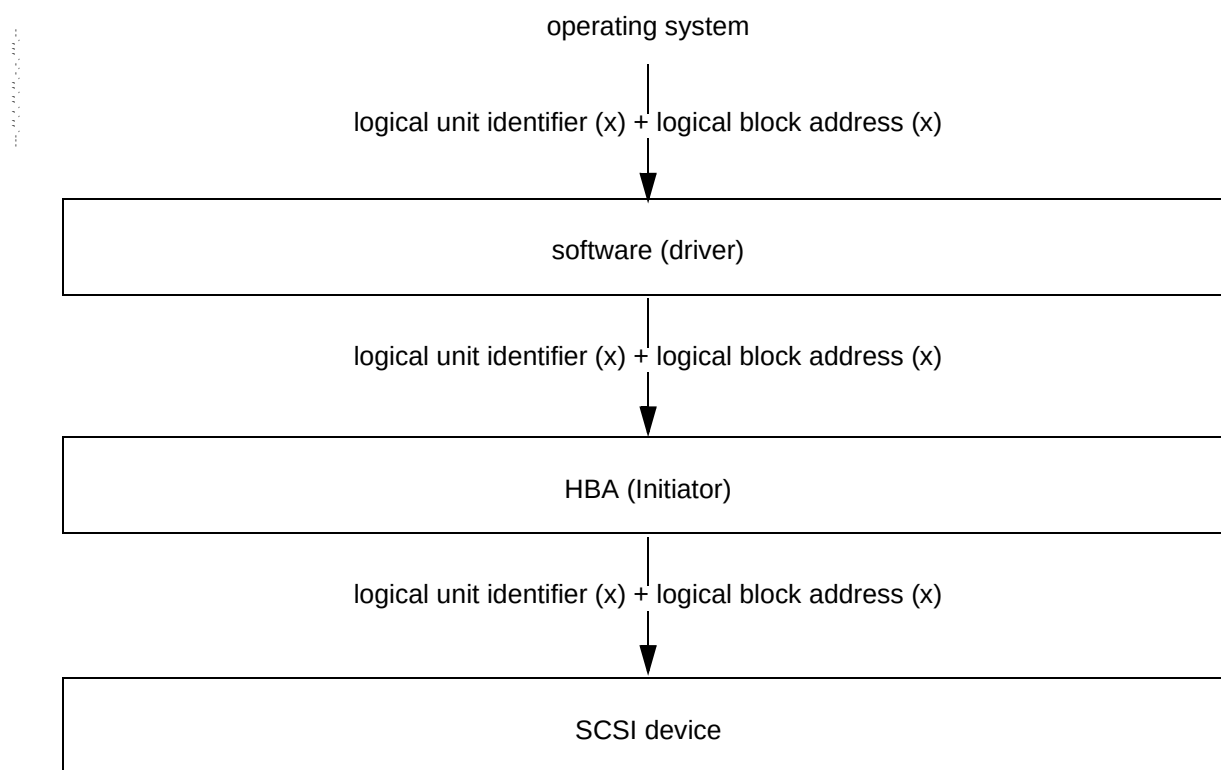


Figure 4 - Branch of generic layers

5.2.6 Software SACL with a branch of SCSI disks

Figure 5 shows system software performing SACL functions. These functions convert the input logical unit identifier(X) and the input logical block address(X) to the output logical unit identifier(Y) and output logical block address(Y). All other layers pass the logical unit identifier and logical block address through.

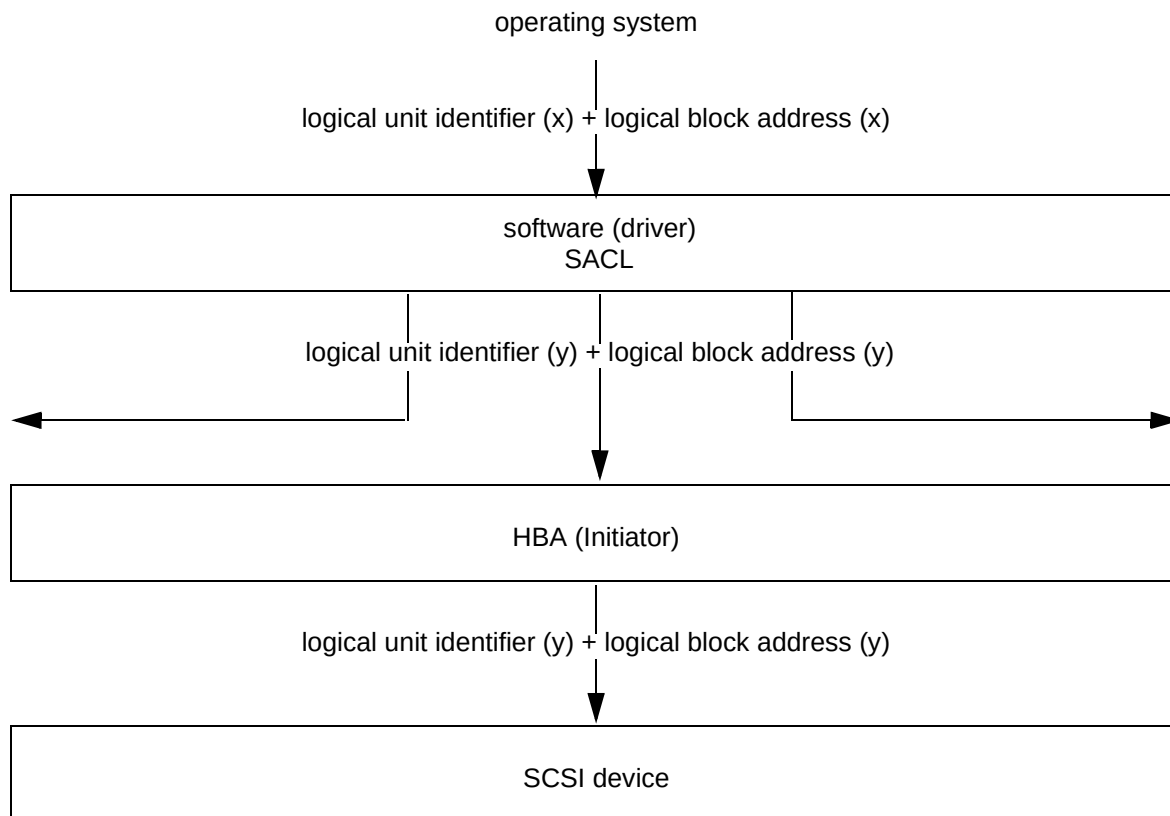


Figure 5 - Software SACL with a branch of SCSI disks

5.2.7 Branch with HBA SACL

Figure 6 shows a HBA performing SACL functions. These functions convert the input logical unit identifier(X) and the input logical block address(X) to the output logical unit identifier(Y) and output logical block address(Y). All other layers pass the logical unit identifier and logical block address through.

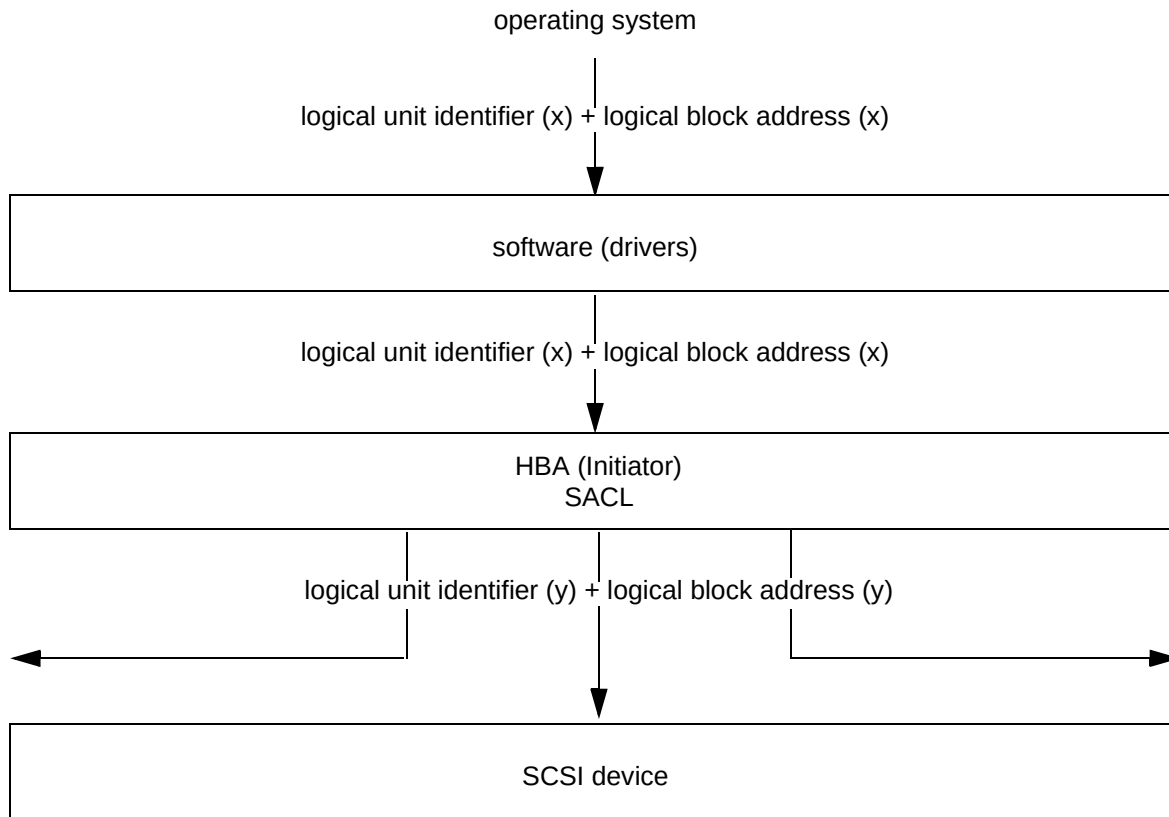


Figure 6 - Branch with HBA SACL

5.2.8 Branch with bridge controller SACL

Figure 7 shows a bridge controller performing SACL functions. These functions convert the input logical unit identifier(X) and the input logical block address(X) to the output logical unit identifier(Y) and output logical block address(Y). All other layers pass the logical unit identifier and logical block address through.

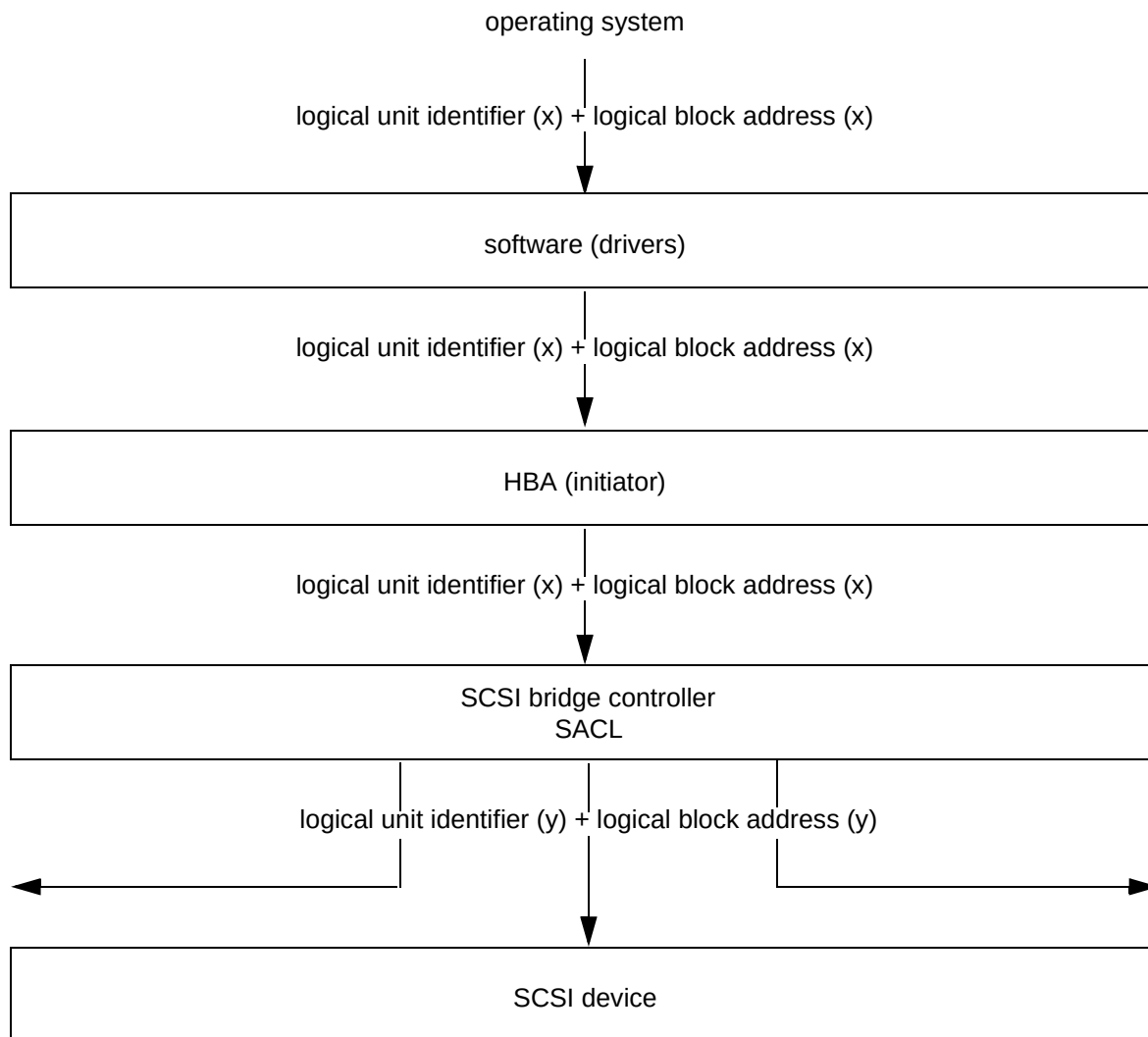


Figure 7 - Branch with bridge controller SACL

5.2.9 Branch with SACs in multiple layers

Figure 8 shows an example with SACLs in multiple layers. This example shows a software layer performing SACL functions and a bridge controller performing SACL functions.

The software functions convert the input logical unit identifier(X) and the input logical block address(X) to the output logical unit identifier(Y) and output logical block address(Y). The bridge controller functions then convert the input logical unit identifier(Y) and the input logical block address(Y) to the output logical unit identifier(Z) and output logical block address(Z). All other layers pass the logical unit identifier and logical block address through.

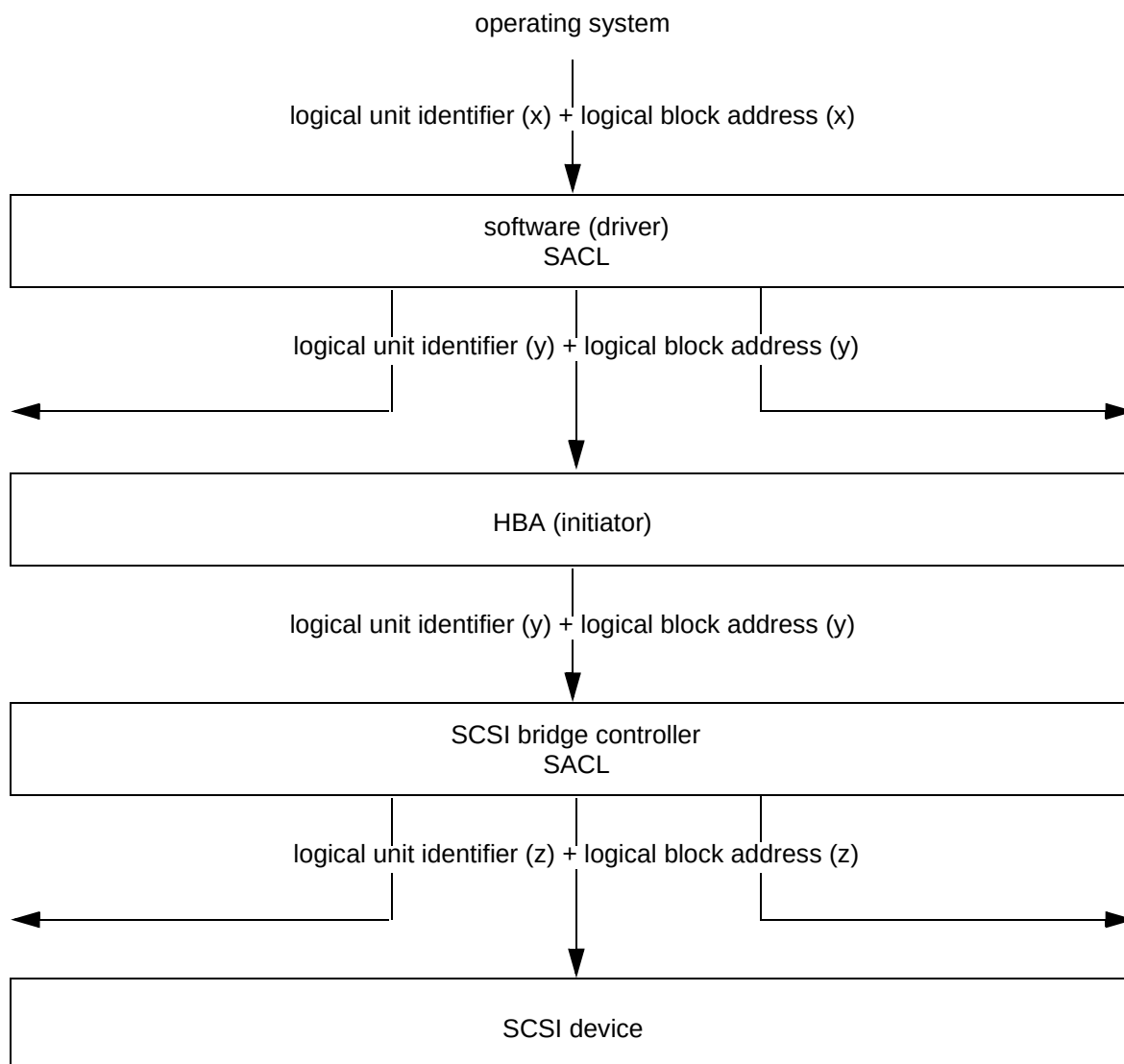


Figure 8 - Branch with SACs in multiple layers

5.3 Model for SCSI-3 storage arrays

The SCSI-3 storage array model defines:

- a) the addressing of multiple devices and the addressing of multiple types of devices through multiple layers of SCSI-3 storage arrays as a single target,
- b) objects and how they are to be configured to create an operational SCSI-3 storage array; and
- c) the operations that occur within SCSI-3 storage arrays.

5.3.1 SCSI storage array addressing

See table 1 for the methods used when addressing a SCSI-3 storage array.

Table 1 - Addressing methods within a SCSI-3 storage array

Method of addressing	Directly addressable by application client	Suffix	Space addressed
Component device	no	_C	none
Peripheral device	yes	_P	physical
Redundancy group	no	_R	protected
Spare	no	_S	none
Volume set	yes	V	user data

The application client shall access redundancy groups, spares, and components by issuing commands to the SCSI-3 storage array's base address (LUN 0). The application client may directly address peripheral devices and volume sets by using eight byte LUN fields (see 5.3.1.2). For SCSI-3 storage array devices that conform to the SCSI-3 Interlocked Protocol Standard (ISO/IEC IS 14776-211) a method has been defined using a modified IDENTIFY message (see A.2) and a LUN mapping page (see 6.10.1.1) to allow direct addressing of peripheral devices and volume sets that conform to the SCSI-3 storage array addressing model.

All peripheral device addresses, except the base address, default to vendor specific values. All component device, redundancy group, spare, and volume set addresses may default to vendor specific values or may be defined by an application client during configuration.

5.3.1.1 SCSI-3 storage array base address

All SCSI-3 storage arrays shall accept a LUN value of zero as a valid address. For SCSI-3 storage arrays that have more than one logical unit, logical unit zero shall be the logical unit that an application client addresses to configure an SCSI-3 storage array and to determine information about the target and the logical units contained within the target. INQUIRY commands to an SCSI-3 storage array base address shall return a device type of array controller device.

To address the base address of the SCSI-3 storage array the peripheral device address method shall be used.

5.3.1.2 Eight byte LUN structure

The eight byte LUN structure (see table 3) allows up to four levels of devices to be addressed under a single target. Each level shall use bytes 0 to 1 to define the address and/or location of the SCSI device to be addressed on that level.

The eight byte LUN structure defined in this standard shall apply to SCSI-3 storage array devices that conform to the SCSI-3 Controller Command.

If the LUN indicates that the command is to be passed to the next layer then the current layer shall use bytes 0 to 1 of the eight byte LUN structure to determine the address of the device to which the command is to be sent. When the command is sent to the target the eight byte LUN structure that was received shall be adjusted to create a new eight byte LUN structure (see table 2).

SCSI-3 storage arrays shall keep track of the necessary addressing information to allow reconnection to the correct task during reselection.

Table 2 - Eight byte LUN structure adjustments

Byte position		
Old		New
0 to 1	Moves to	Not used
2 to 3	Moves to	0 to 1
4 to 5	Moves to	2 to 3
6 to 7	Moves to	4 to 5

Bytes six and seven of each new eight byte LUN structure shall be set to zero.

Table 3 - Eight byte LUN structure

Bit Byte	7	6	5	4	3	2	1	0
0	FIRST LEVEL ADDRESSING							
1								
2	SECOND LEVEL ADDRESSING							
3								
4	THIRD LEVEL ADDRESSING							
5								
6	FOURTH LEVEL ADDRESSING							
7								

The FIRST LEVEL ADDRESSING field indicates the first level address of a peripheral device or volume set. See table 4 for a definition of the FIRST LEVEL ADDRESSING field.

The SECOND LEVEL ADDRESSING field indicates the second level address of a peripheral device or volume set. See table 4 for a definition of the SECOND LEVEL ADDRESSING field.

The THIRD LEVEL ADDRESSING field indicates the third level address of a peripheral device or volume set. See table 4 for a definition of the THIRD LEVEL ADDRESSING field.

The FOURTH LEVEL ADDRESSING field indicates the fourth level address of a peripheral device or volume set. See table 4 for a definition of the FOURTH LEVEL ADDRESSING field.

Table 4 - FIRST LEVEL ADDRESSING field, SECOND LEVEL ADDRESSING field, THIRD LEVEL ADDRESSING field, and FOURTH LEVEL ADDRESSING field

Bit Byte	7	6	5	4	3	2	1	0
n-1	ADDRESS METHOD		ADDRESS METHOD SPECIFIC					
n								

The ADDRESS METHOD field defines the contents of the ADDRESS METHOD SPECIFIC field. See table 5 for the defined address methods.

Table 5 - ADDRESS METHOD

Codes	Description
10b	Logical unit addressing method
00b	Peripheral device addressing method
01b	Volume set addressing method
11b	Reserved

See 5.3.1.5 and 5.3.1.8 for the definitions of the ADDRESS METHOD SPECIFIC field.

5.3.1.3 Component device address method

A component shall only be addressed by addressing the base address of the SCSI-3 storage array that controls or will control the component device. For information on addressing the base address see 5.3.1.1.

5.3.1.4 Logical unit address method

All SCSI commands are allowed when the logical unit address method is selected, however logical units and SCSI-3 storage arrays are only required to support mandatory SCSI commands. SCSI-3 storage arrays are not required to honor pass-through requests from the application client. Any command that is not supported or passed-through shall be terminated with a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID COMMAND OPERATION CODE.

If the logical unit addressing method is selected the SCSI-3 storage array shall relay the received command, if supported, to the addressed logical unit. See table 6 for the definition of the ADDRESS METHOD SPECIFIC field used when the logical unit addressing method is selected.

Table 6 - Logical unit addressing

Bit Byte	7	6	5	4	3	2	1	0
n-1	1	0	TARGET					
n	BUS			LUN				

The TARGET field indicates the target address of the peripheral device to which the SCSI-3 storage array shall relay the received command. The TARGET field indicates the address of the target on the bus indicated by the BUS NUMBER field that the received command shall be relayed to.

NOTE 3 - The value of targets within the TARGET field are defined by individual standards. (e.g., SCSI-3 Parallel Interface Standard (ISO/IEC IS 14776-111) defines targets to be in the range 0 to 7, 0 to 15, and 0 to 31).

The BUS NUMBER field indicates the location of the bus that the SCSI-3 storage array shall use to relay the received command.

The LUN field indicates the address of the logical unit to which the SCSI-3 storage array shall relay the received command. The LUN field indicates the address of the logical unit within the target indicated by the TARGET field that the received command shall be relayed to.

5.3.1.5 Peripheral device address method

All SCSI commands are allowed when the peripheral address method is selected, however peripheral devices and SCSI-3 storage arrays are only required to support mandatory SCSI commands. SCSI-3 storage arrays are not required to honor pass-through requests from the application client. Any command that is not supported or passed-through shall be terminated with a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense code shall be set to INVALID COMMAND OPERATION CODE.

If the peripheral device addressing method is selected the SCSI-3 storage array shall relay the received command, if supported, to the addressed peripheral device providing a bypass of the SACL. See table 7 for the definition of the ADDRESS METHOD SPECIFIC field used when the peripheral device addressing method is selected.

Table 7 - Peripheral device addressing

Bit Byte	7	6	5	4	3	2	1	0
n-1	0	0	BUS NUMBER					
n	TARGET/LUN							

The BUS NUMBER field indicates the location of the bus that the SCSI-3 storage array shall use to relay the received command.

The TARGET/LUN field indicates the address of the peripheral device/LUN to which the SCSI-3 storage array shall relay the received command. If the BUS NUMBER field is set to zero the TARGET/LUN field indicates the

address of the logical unit to relay the received command to, is located within the current level of the SCSI-3 storage array. If the bus number field is not zero the TARGET/LUN field indicates the address of the target on the bus indicated by the bus number field that the received command shall be relay to. The logical unit to relay the received command to shall be LUN zero.

The base device located within the current level shall be addressed by a BUS NUMBER field and a TARGET/LUN field of all zeros.

NOTE 4 - The value of targets within the TARGET/LUN field are defined by individual standards. (e.g., SCSI-3 Parallel Interface Standard (ISO/IEC IS 14776-111) defines targets to be in the range 0 to 7, 0 to 15, and 0 to 31).

5.3.1.6 Redundancy group address method

A redundancy group shall only be addressed by addressing the base address of the SCSI-3 storage array that controls or will control the redundancy group. For information on addressing the base address see 5.3.1.1.

5.3.1.7 Spare address method

A spare shall only be addressed by addressing the base address of the SCSI-3 storage array that controls or will control the spare. For information on addressing the base address see 5.3.1.1.

5.3.1.8 Volume set address method

The volume set address method points to the SACL that executes the command using the algorithms defined by the configuration.

NOTE 5 - The volume set might not be under the control of the addressed SCSI-3 storage array. It is allowed to be in an SCSI-3 storage array lower in the tree structure.

All SCSI commands are allowed when the volume set address method is used, however volume sets are not required to support all SCSI commands. Any command that is not supported shall terminate in error.

In the response to an INQUIRY command the addressed volume set shall return a valid SCSI-3 peripheral device type (e.g., direct access device, streaming device, etc.).

When the volume set addressing method is selected the SCSI-3 storage array shall use a SACL at the current level to address peripheral devices as required to execute the received command. See table 8 for the definition of the ADDRESS METHOD SPECIFIC field used when the volume set addressing method is selected.

Table 8 - Volume set addressing

Bit Byte	7	6	5	4	3	2	1	0
n-1	0	1	(MSB)					
n	LUN							(LSB)

The LUN field indicates the address of the volume set that the SCSI-3 storage array shall direct the received command to.

5.3.2 SCSI-3 storage array objects

An SCSI-3 storage array consists of several objects that are configured to create an operating device. The rules for configuring objects and their interactions are defined in the following clauses. An SCSI-3 storage array shall save the current configuration throughout all task management functions and power conditions.

NOTE 6 - The configuration becomes the current configuration on successful completion of a configuration command.

The objects are:

- a) component device;
- b) peripheral device;
- c) p_extent;
- d) ps_extent;
- e) redundancy group;
- f) spare;
- g) volume set.

5.3.2.1 Adding objects

Objects that have been added to a SCSI-3 storage array shall be addressable (see 5.3.1) by an application client.

Peripheral devices and component devices may be logically added to a SCSI-3 storage array. The ADD PERIPHERAL DEVICE/COMPONENT DEVICE service action (see 6.3.1.1) adds objects. Adding a peripheral device or a component device may or may not automatically cause the device to become part of a configuration.

5.3.2.2 Association of objects

Objects become associated during the configuration of SCSI-3 storage arrays. The create/modify service actions for redundancy groups (see 6.5.1.2) and volume sets (see 6.7.1.3) create associations and the delete service actions (see 6.5.1.3 and 6.7.1.4) remove associations. The rules for objects that are associated are explicitly defined throughout this clause and within the create/modify service actions.

An SCSI-3 storage array may create associations through mechanisms other than the create/modify service actions defined in this document (e.g., by factory settings or by external configuration facilities). Whether such associations may be modified by the create/modify service actions specified by this standard or by other mechanisms is vendor specific. The only requirement on such associations is that they shall be reported using the report service actions defined in this standard.

NOTE 7 - One example of an association would be to associate a group of p_extents with a redundancy group using a CREATE/MODIFY REDUNDANCY GROUP service action.

5.3.2.3 Attachment of objects

Objects shall only be attached to component devices. The ATTACH TO COMPONENT DEVICE service action (6.3.1.2) establishes attachments.

The behavior of attachments and their interactions with component devices are vendor specific.

The attachment of component devices shall not be reflexive (i.e. attaching LUN_C 1 to LUN_C 2 does not attach LUN_C 2 to LUN_C 1).

NOTE 8 - One example of an attachment would be to attach a redundancy group to a specific power supply. This attachment could mean that the power supply would only supply power to that redundancy

group. Another example would be to attach a controller card to another controller card. This attachment could mean that the work load is shared between the two controllers.

An SCSI-3 storage array may create attachments through mechanisms other than the attach service actions defined in this document (e.g., by factory settings or by external configuration facilities). Whether such attachments may be modified by the attach service actions specified by this standard or by other mechanisms is vendor specific. The only requirement on such attachments is that they shall be reported using the report service actions defined in this standard.

5.3.2.4 Covering of objects

Any object or group of objects may be covered by a like object. The SPARE(OUT) command (see 6.9.1) establishes the covering of the objects.

When an object or group of objects are covered and there is a failure, the covering object shall assume all the characteristics of the failed object, in effect doing an automatic exchange. The length of time the exchange stays in effect is vendor specific.

Objects may be covered by a SCSI-3 storage array without any spare requests being issued from an application client. The SCSI-3 storage array may have covered objects based on internal requirements.

NOTE 9 - One example of covering would be for a peripheral device to cover like peripheral device(s) within a redundancy group. This would cause any peripheral device that failed within the covered redundancy group to be replaced by the peripheral device covering that redundancy group. Two controllers could also be configured to cover one another, using two spare service actions, in which case if either fails the other will take over the entire work load.

5.3.2.5 Exchanging objects

A peripheral device, p_extent, or component device may be exchanged with a like device. The exchange service actions (see 6.3.1.4 and 6.3.1.5) are used to exchange objects. When an old peripheral device, p_extent, or component device is exchanged, the new peripheral device, p_extent, or component device takes on all the characteristics of the old peripheral device, p_extent, or component device (e.g., redundancy group, volume set mappings, attachments, etc.). An exact copy of any protected space contents and/or check data on the old device shall be replicated on the new device. The characteristics of the old peripheral device are vendor specific.

5.3.2.6 Protected objects

Protected objects are objects that are able to tolerate one or more objects failing without any loss of user data or loss of SCSI-3 storage array availability.

5.3.2.7 Removing objects

Objects that have been removed from a SCSI-3 storage array shall not be addressable by an application client, until the object is re-added or recreated. The remove service actions (see 6.3.1.7, 6.5.1.3 and 6.9.1.3) are used to remove objects.

Any object may be removed from a configuration. However, some restrictions may apply as to when an object may be removed. Those restrictions are defined in 6.3.1.7, 6.5.1.3, and 6.9.1.3.

When the object is removed all attachments, associations, and covers relating to the object being removed are erased from any configurations containing the object. Any addressable logical block addresses within a removed volume set shall become unassigned protected space. Any addressable logical block addresses within a removed redundancy group or spare shall become unassigned p_extent space.

5.3.2.8 Component device

See 3.1.10 for the definition of component devices.

5.3.2.9 Peripheral device

See 3.1.26 for the definition of peripheral devices.

5.3.2.10 P_extent

See 3.1.25 for the definition of p_extents.

P_extents are used by the application client to create and modify redundancy groups and spares. A single assigned p_extent or unassigned p_extent that contains no check data may be configured into one or more redundancy groups. A single assigned p_extent or unassigned p_extent may be configured into one or more spares. A single p_extent shall not be associated with a redundancy group and configured as a spare at the same time. After a p_extent has been configured into a redundancy group or as a spare it becomes an assigned p_extent.

Any addressable logical block addresses within an operating peripheral device connected within a SCSI-3 storage array that have not been configured into a redundancy group shall be unassigned p_extents. Before assignment, all the consecutive addressable logical block addresses on a single peripheral device shall be grouped into a single unassigned p_extent.

The application client uses a REPORT ASSIGNED/UNASSIGNED P_EXTENT service action (see 6.2.1.1) to determine the unassigned p_extents.

A single unassigned p_extent may be subdivided into one or more p_extents by the application client. The subdivided p_extents may be used within one or more CREATE/MODIFY REDUNDANCY GROUP service actions.

5.3.2.11 Ps_extent

See 3.1.28 for the definition of ps_extent.

Ps_extents are used by the application client to create and modify volume sets. Ps_extents are created on the successful completion of a CREATE/MODIFY REDUNDANCY GROUP service action (6.5.1.2.) A single ps_extent shall only be configured into one volume set. After a ps_extent has been configured into a volume set it becomes an assigned ps_extent.

Any addressable logical block addresses, not defined as containing check data, within an operating peripheral device connected within a SCSI-3 storage array that have not been configured into a volume set shall be unassigned ps_extents. All the consecutive addressable logical block addresses on a single peripheral device, excluding logical blocks defined as containing check data, shall be grouped into a single unassigned ps_extent.

The application client uses a REPORT UNASSIGNED REDUNDANCY GROUP SPACE service action (see 6.4.1.2) to determine the unassigned ps_extents.

A single unassigned ps_extent may be subdivided into one or more ps_extents by the application client. The subdivided ps_extents may be used within one or more CREATE/MODIFY VOLUME SET service actions.

A ps_extent is different from a p_extent in that the ps_extent does not include any logical block addresses that have been mapped as check data where the p_extent includes all addressable logical blocks within the selected range on a peripheral device.

5.3.2.12 Redundancy group

See 3.1.31 for the definition of redundancy group.

Redundancy groups are created by the application client to protect user data contained within volume sets by mapping check data within a redundancy group. Redundancy groups may overlap, however, the underlying p_extents within the overlap shall not contain any check data.

As a result of a successful creation of a redundancy group ps_extents are formed. The ps_extents then may be used by the application client to create volume sets.

Figure 9 shows the relationship between the check data and protected space before any volume sets have been defined.

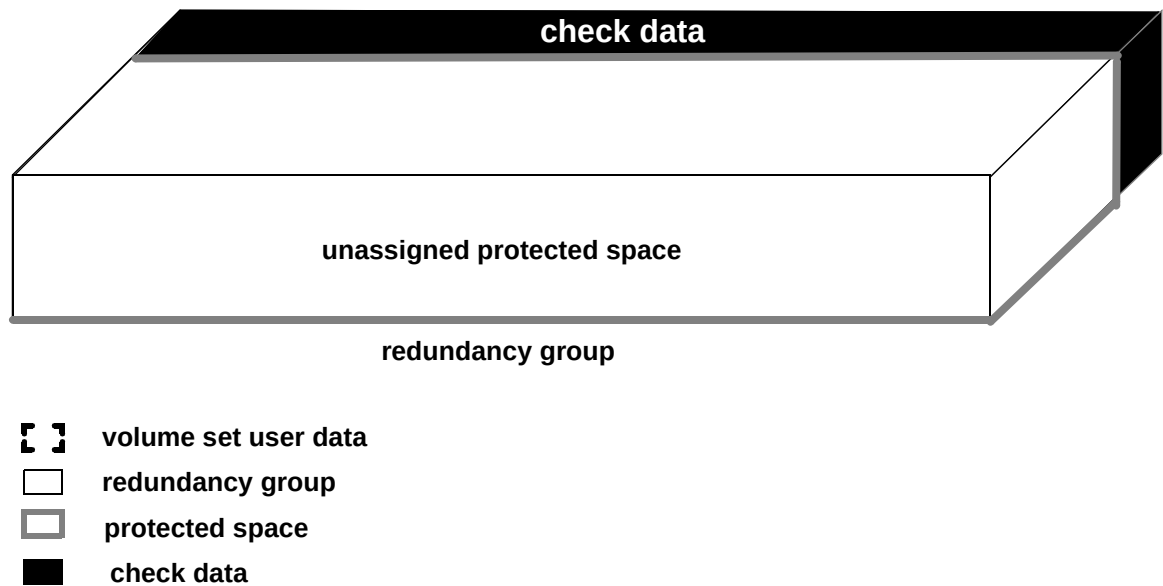


Figure 9 - Single redundancy group

Figure 10 shows the relationship between the check data and protected space after two volume sets have been defined within the redundancy group.

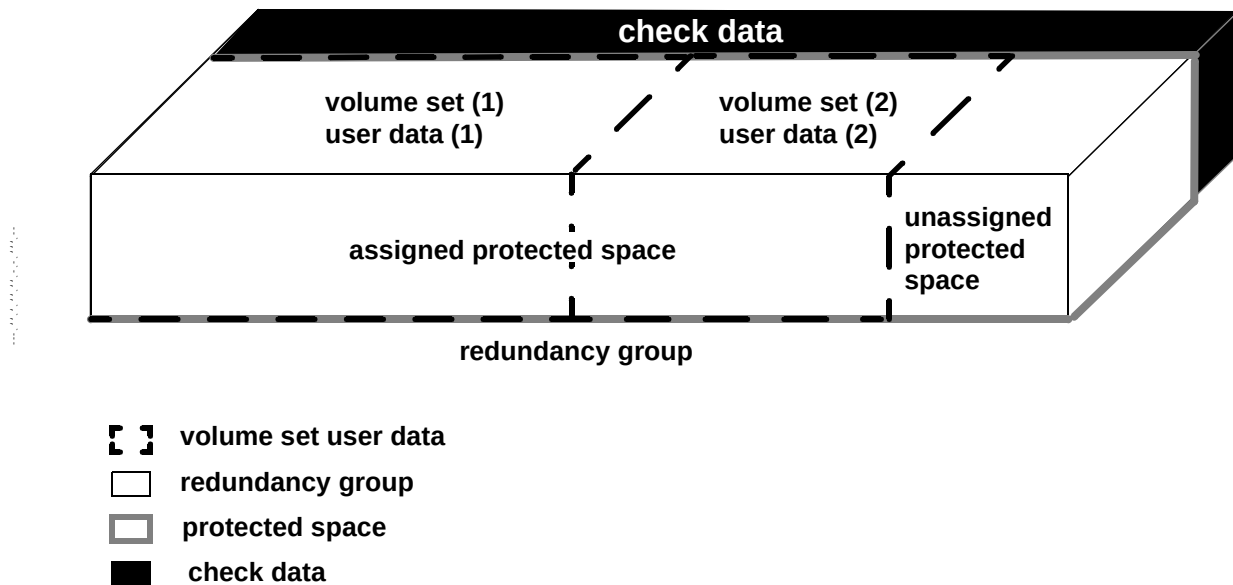


Figure 10 - Multiple volume sets associated with a single redundancy group

The application client may map check data within a redundancy group using the following three parameters:

- a) START CHECK DATA INTERLEAVE UNIT field;
- b) NUMBER OF UNITS OF CHECK DATA field;
- c) NUMBER OF UNITS OF USER DATA field.

The START CHECK DATA INTERLEAVE UNIT field contains the location of the first unit of check data within the p_extent. There is a unique START CHECK DATA INTERLEAVE UNIT field for each p_extent.

The NUMBER OF UNITS OF CHECK DATA field contains the number of consecutive units to be reserved for check data within the p_extent. There is a unique NUMBER OF UNITS OF CHECK DATA field for each p_extent.

The NUMBER OF UNITS OF USER DATA field contains the number of consecutive units to be reserved for protected space within the p_extent. There is a unique NUMBER OF UNITS OF USER DATA field for each p_extent.

The flow chart in figure 11 is an example of an implementation of a SCSI-3 storage array redundancy group being configured with a check data mapping of XOR redundancy. The check data mapping routine uses the START CHECK DATA INTERLEAVE UNIT field, NUMBER OF UNITS OF CHECK DATA field, and NUMBER OF UNITS OF USER DATA field and assumes a non-zero value for the NUMBER OF UNITS OF CHECK DATA field and the NUMBER OF UNITS USER DATA field.

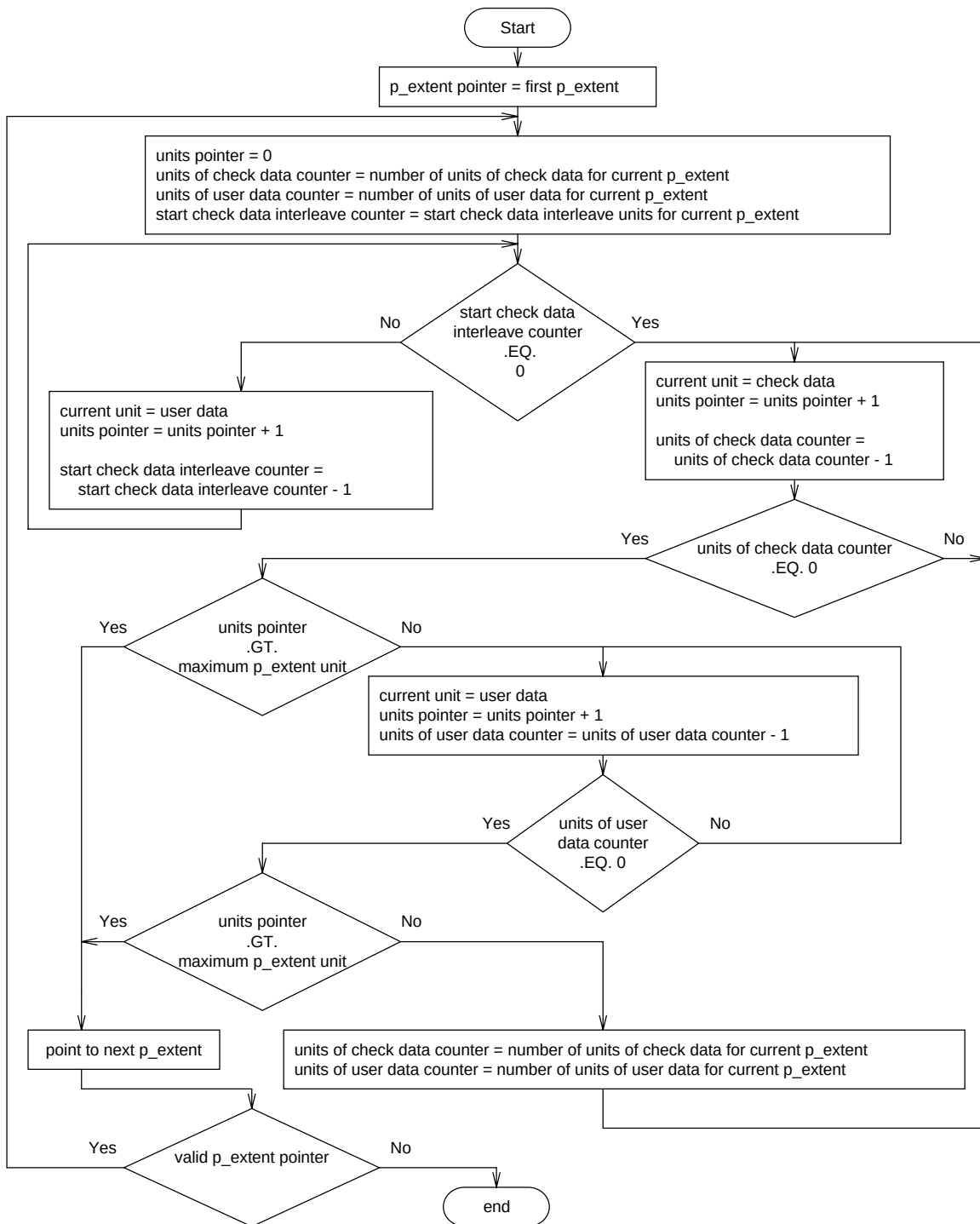


Figure 11 - Redundancy group check data mapping flow chart

The application may choose any one of the following methods of check data mapping within a redundancy group:

a) No redundancy;

- b) Copy redundancy;
- c) XOR redundancy;
- d) P+Q redundancy;
- e) Vendor specific redundancy.

5.3.2.12.1 No redundancy method of check data mapping

In a redundancy group that the application client requests no redundancy, user data is not protected. If this method is requested then the following fields in the CREATE/MODIFY REDUNDANCY GROUP service action (see 6.5.1.2) shall all be set as indicated:

- a) GRANULARITY OF UNITS field = 0;
- b) START CHECK DATA INTERLEAVE UNIT field = 0;
- c) NUMBER OF UNITS OF CHECK DATA field = 0;
- d) NUMBER OF UNITS OF USER DATA field = set to desired value.

5.3.2.12.2 Copy redundancy method of check data mapping

In a redundancy group that the application client requests copy redundancy all user data is replicated on all the p_extents listed in the CREATE/MODIFY REDUNDANCY GROUP service action (6.5.1.2). If this method is requested then the following fields in the CREATE/MODIFY REDUNDANCY GROUP service action (see 6.5.1.2) shall be set as indicated:

- a) GRANULARITY OF UNITS field = logical block;
- b) START CHECK DATA INTERLEAVE UNIT field = 0;
- c) NUMBER OF UNITS OF CHECK DATA field = 0;
- d) NUMBER OF UNITS OF USER DATA field = set to desired value.

For copy redundancy the NUMBER OF UNITS OF USER DATA field shall be equal for all the p_extents within a single CREATE/MODIFY REDUNDANCY GROUP service action.

5.3.2.12.3 XOR or P+Q redundancy method of check data mapping

In a redundancy group that the application client requests XOR redundancy, or P+Q redundancy the user data is protected by use of check data located within the check data areas. The check data mapping is defined by the check data interleave fields. If this method is requested then the following fields in the CREATE/MODIFY REDUNDANCY GROUP service action (see 6.5.1.2) shall be set as indicated:

- a) GRANULARITY OF UNITS field = set to desired value;
- b) START CHECK DATA INTERLEAVE UNIT field = set to desired value;
- c) NUMBER OF UNITS OF CHECK DATA field = set to desired value;
- d) NUMBER OF UNITS OF USER DATA field = set to desired value.

5.3.2.12.4 Vendor specific redundancy method of check data mapping

In a redundancy group for which the application client requests a redundancy type method of vendor specific redundancy, the user data is protected in a vendor specific manner.

5.3.2.13 Spares

See 3.1.37 for the definition of spares.

Spares are created by the application client to allow a failed p_extent, peripheral device, or component device to be automatically exchanged (see 5.3.2.5). P_extents shall only be exchanged with p_extents, peripheral devices shall only be exchanged with peripheral devices, and component devices shall only be exchanged with component devices. A redundancy group or peripheral device that contains a failed range of LBA_P(s), a failed peripheral device, or a failed component shall be covered by a spare before an

automatic exchange is allowed to occur.

After an automatic exchange the spare takes on all the characteristics of the failed p_extent, peripheral device, or component device. After the automatic exchange the covering p_extent, peripheral device, or component device shall no longer be available to cover another object. The failed p_extent, peripheral device, or component device shall be marked as broken.

The method for replacing the failed p_extent, peripheral device, or component device and restoring the spare is vendor specific.

NOTE 10 - One method of replacing a failed p_extent, peripheral device, or component device would be to automatically exchange a new p_extent, peripheral device, or component device with the spare that covered the failure. After a successful exchange the spare is available to cover failures and the failed p_extent, peripheral device, or component device is as it was before the failure.

NOTE 11 - Another method would be to automatically add the new p_extent, peripheral device, or component device into the SCSI-3 storage array, delete the original covering spare, then create a new spare with the new p_extent, peripheral device, or component device that would cover the same p_extent, peripheral device, or component device as the original spare.

5.3.2.14 Volume sets

See 3.1.47 for the definition of volume sets.

Volume sets are created by the application client to provide a contiguous range of logical block addresses for reading and writing user data. Volume sets shall not overlap and shall be independent from one another.

The application client issues a REPORT UNASSIGNED REDUNDANCY GROUP SPACE service action (see 6.4.1.2) to determine the boundaries of any unassigned protected space. The unassigned protected space may be used to create volume sets or to expand existing volume sets. This service action provides information to the application client so it can place volume sets on specific peripheral devices.

Figure 12 shows a relationship between the check data and protected space before any volume sets have been defined.

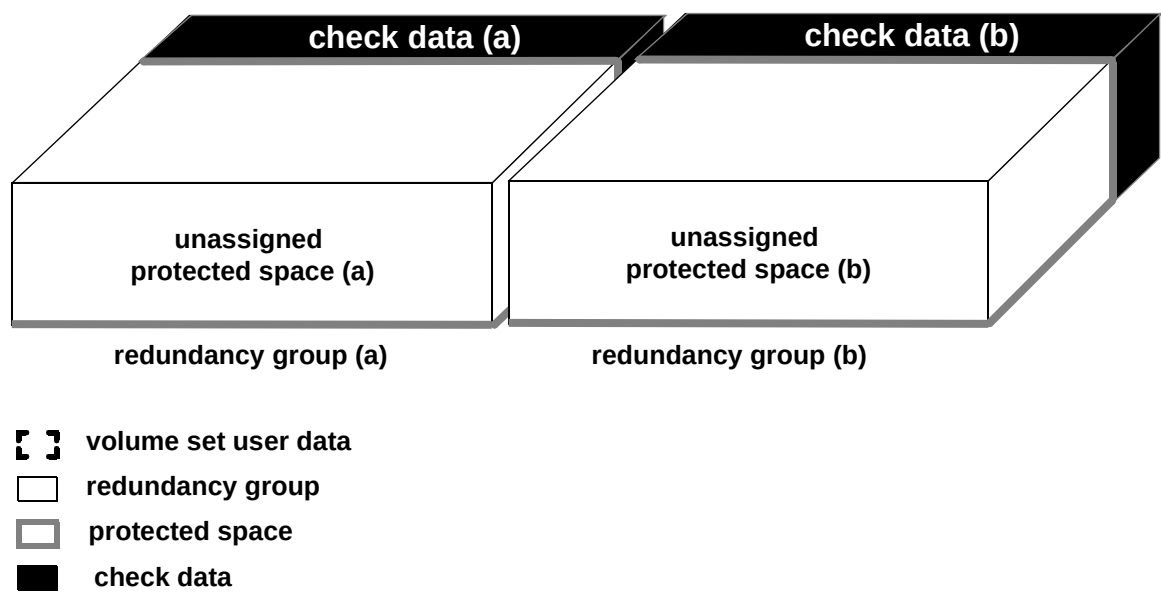


Figure 12 - Multiple redundancy groups

Figure 13 shows the relationship between check data, user data and protected space after a single volume set has been defined across the two redundancy groups.

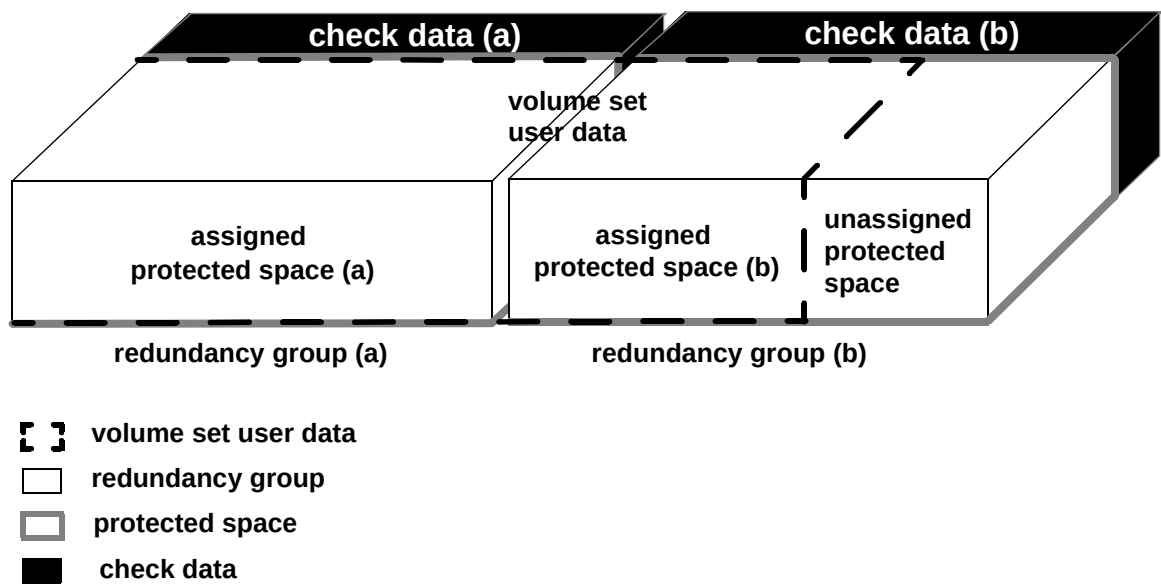


Figure 13 - Single volume set associated with multiple redundancy groups

NOTE 12 - The volume set in figure 13 is ready to receive user data.

The application client may map user data within a volume set using the following three parameters:

- a) USER DATA STRIPE DEPTH field;
- b) PS_EXTENT STRIPE LENGTH field;
- c) PS_EXTENT INTERLEAVE DEPTH field.

The USER DATA STRIPE DEPTH field contains the number of contiguous units to count within a ps_extent before proceeding to the next ps_extent. There is a USER DATA STRIPE DEPTH field for each ps_extent.

The PS_EXTENT STRIPE LENGTH field contains the number of contiguous ps_extents to count before looping back to the first ps_extent of the current stripe.

The PS_EXTENT INTERLEAVE DEPTH field contains the number of stripes to count before continuing onto the next consecutive ps_extent beyond the current stripe.

The ps_extent interleave depth is only used if the ps_extent stripe length is not equal to the number of ps_extents.

The flow chart in figure 14 and figure 15 is an example of an implementation of a SCSI-3 storage array volume set user data mapping routine using the USER DATA STRIPE DEPTH field, PS_EXTENT STRIPE length field, and ps_extent interleave depth field. For simplicity the user data stripe depth field is assumed to be the same value for all the ps_extents in the flow chart.

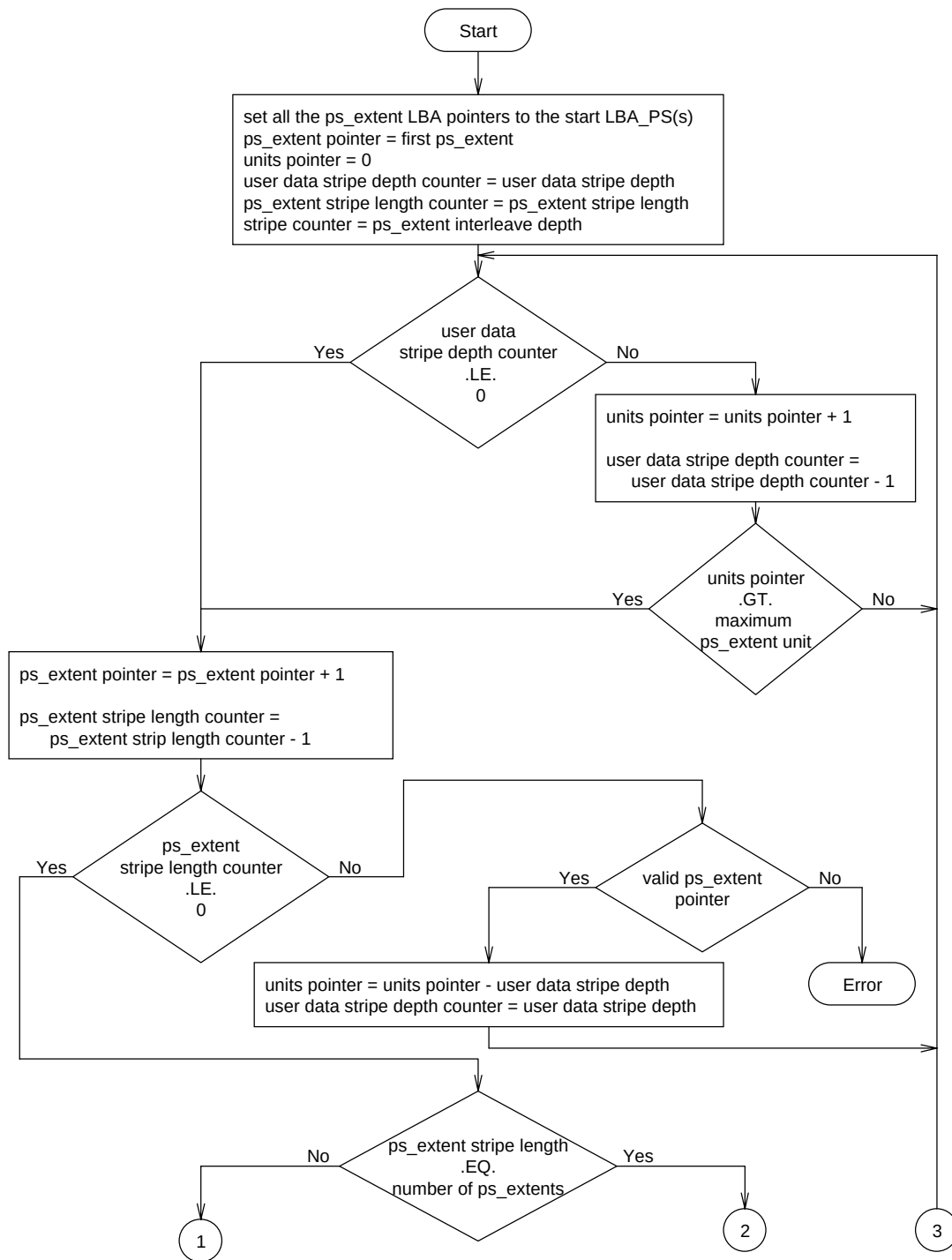


Figure 14 - Volume set user data mapping flow chart (part 1)

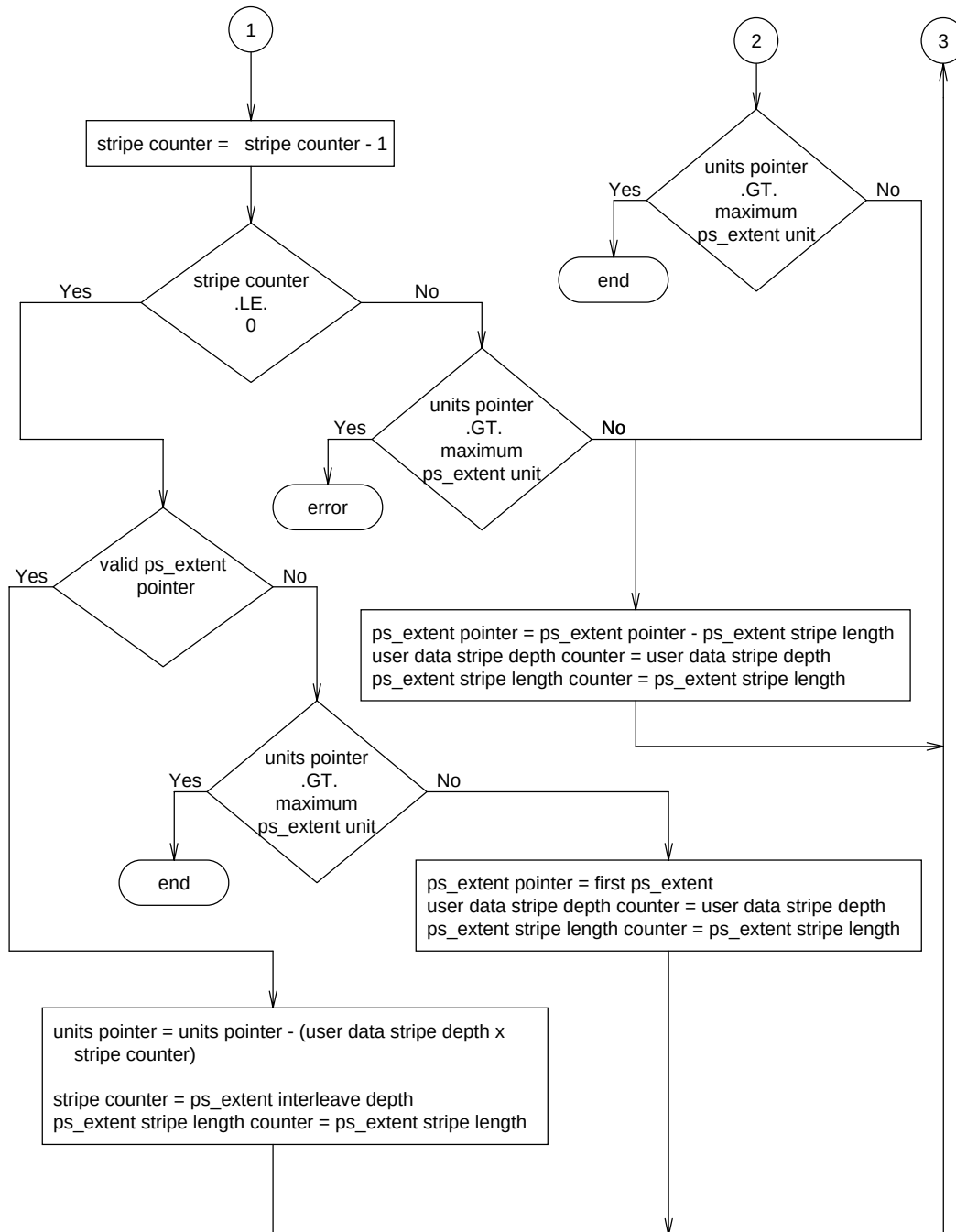


Figure 15 - Volume set user data mapping flow chart (part 2)

Figure 16 shows the most general implementation of the three parameters used to map the user data.