



1
2
3
4

Document Number: DSP0838

Date: 2009-06-04

Version: 1.0.0

5
6

PCI Device Profile SM CLP Command Mapping Specification

7 **Document Type: Specification**
8 **Document Status: DMTF Standard**
9 **Document Language: E**

10

11 Copyright notice

12 Copyright © 2006, 2009 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

13 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
14 management and interoperability. Members and non-members may reproduce DMTF specifications and
15 documents, provided that correct attribution is given. As DMTF specifications may be revised from time to
16 time, the particular version and release date should always be noted.

17 Implementation of certain elements of this standard or proposed standard may be subject to third party
18 patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations
19 to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose,
20 or identify any or all such third party patent right, owners or claimants, nor for any incomplete or
21 inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to
22 any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize,
23 disclose, or identify any such third party patent rights, or for such party's reliance on the standard or
24 incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any
25 party implementing such standard, whether such implementation is foreseeable or not, nor to any patent
26 owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is
27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party
28 implementing the standard from any and all claims of infringement by a patent owner for such
29 implementations.

30 For information about patents held by third-parties which have notified the DMTF that, in their opinion,
31 such patent may relate to or impact implementations of DMTF standards, visit
32 <http://www.dmtf.org/about/policies/disclosures.php>.

33

CONTENTS

34 Foreword 5
 35 Introduction 6
 36 1 Scope 7
 37 2 Normative References..... 7
 38 2.1 Approved References 7
 39 2.2 Other References..... 7
 40 3 Terms and Definitions..... 7
 41 4 Symbols and Abbreviated Terms..... 8
 42 5 Recipes..... 9
 43 6 Mappings..... 9
 44 6.1 CIM_ElementCapabilities 9
 45 6.2 CIM_ControlledBy..... 12
 46 6.3 CIM_DeviceConnection 14
 47 6.4 CIM_HostedCollection 16
 48 6.5 CIM_ConcreteIdentity 18
 49 6.6 CIM_MemberOfCollection 20
 50 6.7 CIM_SystemDevice 22
 51 6.8 CIM_PCIPort..... 25
 52 6.9 CIM_PCIPortGroup..... 28
 53 6.10 CIM_PCIDevice 30
 54 6.11 CIM_PCIBridge 35
 55 6.12 CIM_PCleSwitch..... 40
 56 ANNEX A (informative) Change Log 47
 57

58 Tables

59 Table 1 – Command Verb Requirements for CIM_ElementCapabilities 10
 60 Table 2 – Command Verb Requirements for CIM_ControlledBy..... 12
 61 Table 3 – Command Verb Requirements for CIM_DeviceConnection 14
 62 Table 4 – Command Verb Requirements for CIM_HostedCollection 16
 63 Table 5 – Command Verb Requirements for CIM_ConcreteIdentity 18
 64 Table 6 – Command Verb Requirements for CIM_MemberOfCollection 20
 65 Table 7 – Command Verb Requirements for CIM_SystemDevice 23
 66 Table 8 – Command Verb Requirements for CIM_PCIPort..... 26
 67 Table 9 – Command Verb Requirements for CIM_PCIPortGroup..... 28
 68 Table 10 – Command Verb Requirements for CIM_PCIDevice 30
 69 Table 11 – Command Verb Requirements for CIM_PCIBridge 35
 70 Table 12 – Command Verb Requirements for CIM_PCleSwitch..... 40
 71

73

Foreword

74 The *PCI Device Profile SM CLP Command Mapping Specification* (DSP0838) was prepared by the
75 Server Management Working Group.

76 **Conventions**

77 The pseudo-code conventions utilized in this document are the Recipe Conventions as defined in SNIA
78 [SMI-S 1.1.0](#), section 7.6.

79 **Acknowledgements**

- 80 • Ravi Mantena – HP
- 81 • Aaron Merkin – IBM
- 82 • Brady Evans – HP
- 83 • Christina Shaw – HP
- 84 • Jon Hass – Dell
- 85 • Jeff Hilland – HP
- 86 • John Leung – Intel
- 87 • Khachatur Papanyan – Dell

88

89

Introduction

90 This document defines the SM CLP mapping for CIM elements described in the [PCI Device Profile](#). The
91 information in this specification, combined with the *SM CLP-to-CIM Common Mapping Specification 1.0*
92 ([DSP0216](#)), is intended to be sufficient to implement SM CLP commands relevant to the classes,
93 properties, and methods described in the [PCI Device Profile](#) using CIM operations.

94 The target audience for this specification is implementers of the SM CLP support for the [PCI Device](#)
95 [Profile](#).

96 PCI Device Profile SM CLP Command Mapping Specification

97 1 Scope

98 This specification contains the requirements for an implementation of the SM CLP to provide access to,
99 and implement the behaviors of, the [PCI Device Profile](#).

100 2 Normative References

101 The following referenced documents are indispensable for the application of this document. For dated
102 references, only the edition cited applies. For undated references, the latest edition of the referenced
103 document (including any amendments) applies.

104 2.1 Approved References

105 DMTF DSP1075, *PCI Device Profile 1.0.0*,
106 http://www.dmtf.org/standards/published_documents/DSP1075_1.0.0.pdf

107 DMTF DSP0216, *SM CLP-to-CIM Common Mapping Specification 1.0.0*,
108 http://www.dmtf.org/standards/published_documents/DSP0216_1.0.0.pdf

109 SNIA, *Storage Management Initiative Specification (SMI-S) 1.1.0*,
110 http://www.snia.org/tech_activities/standards/curr_standards/smi

111 2.2 Other References

112 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
113 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>

114 3 Terms and Definitions

115 For the purposes of this document, the following terms and definitions apply.

116 3.1

117 **can**

118 used for statements of possibility and capability, whether material, physical, or causal

119 3.2

120 **cannot**

121 used for statements of possibility and capability, whether material, physical or causal

122 3.3

123 **conditional**

124 indicates requirements to be followed strictly in order to conform to the document when the specified
125 conditions are met

126 3.4

127 **mandatory**

128 indicates requirements to be followed strictly in order to conform to the document and from which no
129 deviation is permitted

- 130 **3.5**
131 **may**
132 indicates a course of action permissible within the limits of the document
- 133 **3.6**
134 **need not**
135 indicates a course of action permissible within the limits of the document
- 136 **3.7**
137 **optional**
138 indicates a course of action permissible within the limits of the document
- 139 **3.8**
140 **shall**
141 indicates requirements to be followed strictly in order to conform to the document and from which no
142 deviation is permitted
- 143 **3.9**
144 **shall not**
145 indicates requirements to be followed strictly in order to conform to the document and from which no
146 deviation is permitted
- 147 **3.10**
148 **should**
149 indicates that among several possibilities, one is recommended as particularly suitable, without
150 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 151 **3.11**
152 **should not**
153 indicates that a certain possibility or course of action is deprecated but not prohibited

154 **4 Symbols and Abbreviated Terms**

155 The following symbols and abbreviations are used in this document.

- 156 **4.1**
157 **CIM**
158 Common Information Model
- 159 **4.2**
160 **CLP**
161 Command Line Protocol
- 162 **4.3**
163 **DMTF**
164 Distributed Management Task Force
- 165 **4.4**
166 **PCI**
167 Peripheral Component Interconnect

- 168 **4.5**
169 **PCIe**
170 Peripheral Component Interconnect Express
- 171 **4.6**
172 **SM**
173 Server Management
- 174 **4.7**
175 **SMI-S**
176 Storage Management Initiative Specification
- 177 **4.8**
178 **SNIA**
179 Storage Networking Industry Association

180 **5 Recipes**

181 The following is a list of the common recipes used by the mappings in this specification. For a definition of
182 each recipe, see the *SM CLP-to-CIM Common Mapping Specification 1.0* ([DSP0216](#)).

- 183 • smShowInstance()
- 184 • smShowInstances()
- 185 • smSetInstance()
- 186 • smShowAssociationInstance()
- 187 • smShowAssociationInstances()

188 This mapping does not define any recipes for local reuse.

189 **6 Mappings**

190 The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in
191 the [PCI Device Profile](#). Requirements specified here related to support for a CLP verb for a particular
192 class are solely within the context of this profile.

193 **6.1 CIM_ElementCapabilities**

194 The `cd`, `exit`, `help`, and `version`, verbs shall be supported as described in [DSP0216](#).

195 Table 1 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
196 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
197 verb and target. Table 1 is for informational purposes only; in case of a conflict between Table 1 and
198 requirements detailed in the following sections, the text detailed in the following sections supersedes the
199 information in Table 1.

200

Table 1 – Command Verb Requirements for CIM_ElementCapabilities

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.1.2.
start	Not supported	
stop	Not supported	

201 No mapping is defined for the following verbs for the specified target: *create*, *delete*, *dump*, *load*,
 202 *reset*, *set*, *start*, and *stop*.

203 6.1.1 Ordering of Results

204 When results are returned for multiple instances of CIM_ElementCapabilities, implementations shall
 205 utilize the following algorithm to produce the natural (that is, default) ordering:

- 206 • Results for CIM_ElementCapabilities are unordered; therefore, no algorithm is defined.

207 6.1.2 Show

208 This section describes how to implement the *show* verb when applied to an instance of
 209 CIM_ElementCapabilities. Implementations shall support the use of the *show* verb with
 210 CIM_ElementCapabilities.

211 6.1.2.1 Show Command Form for Multiple Instances Target – 212 CIM_EnabledLogicalElementCapabilities Reference

213 This command form is used to show many instances of CIM_ElementCapabilities. This command form
 214 corresponds to a *show* command issued against instances of CIM_ElementCapabilities where only one
 215 reference is specified and the reference is to an instance of CIM_EnabledLogicalElementCapabilities.

216 6.1.2.1.1 Command Form

```
217 show <CIM_ElementCapabilities multiple instances>
```

218 6.1.2.1.2 CIM Requirements

219 See CIM_ElementCapabilities in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
 220 mandatory properties.

221 6.1.2.1.3 Behavior Requirements

222 6.1.2.1.3.1 Preconditions

223 \$instance represents the instance of CIM_EnabledLogicalElementCapabilities which is referenced by
 224 CIM_ElementCapabilities.

225 6.1.2.1.3.2 Pseudo Code

```
226 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );
227 &smEnd;
```

228 6.1.2.2 Show Command Form for a Single Instance – CIM_PCIDevice Reference

229 This command form is used to show a single instance of CIM_ElementCapabilities. This command form
 230 corresponds to a `show` command issued against a single instance of CIM_ElementCapabilities where
 231 only one reference is specified and the reference is to the instance of CIM_PCIDevice.

232 6.1.2.2.1 Command Form

```
233 show <CIM_ElementCapabilities single instance>
```

234 6.1.2.2.2 CIM Requirements

235 See CIM_ElementCapabilities in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
 236 mandatory properties.

237 6.1.2.2.3 Behavior Requirements

238 6.1.2.2.3.1 Preconditions

239 \$instance represents the instance of CIM_PCIDevice which is referenced by CIM_ElementCapabilities.

240 6.1.2.2.3.2 Pseudo Code

```
241 &smShowAssociationInstances ( "CIM_ElementCapabilities", $instance.getObjectPath() );
242 &smEnd;
```

243 6.1.2.3 Show Command Form for a Single Instance Target – Both References

244 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 245 the `show` command issued against CIM_ElementCapabilities where both references are specified and
 246 therefore the desired instance is unambiguously identified.

247 6.1.2.3.1 Command Form

```
248 show <CIM_ElementCapabilities single instance>
```

249 6.1.2.3.2 CIM Requirements

250 See CIM_ElementCapabilities in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
 251 mandatory properties.

252 6.1.2.3.3 Behavior Requirements

253 6.1.2.3.3.1 Preconditions

254 \$instanceA represents the referenced instance of CIM_PCIDevice through the CIM_ElementCapabilities
 255 association.

256 \$instanceB represents the instance of CIM_EnabledLogicalElementCapabilities which is referenced by
 257 CIM_ElementCapabilities.

258 6.1.2.3.3.2 Pseudo Code

```
259 &smShowAssociationInstance ( "CIM_ElementCapabilities", $instanceA.getObjectPath(),
260     $instanceB.getObjectPath() );
261 &smEnd;
```

262 6.2 CIM_ControlledBy

263 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

264 Table 2 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 265 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 266 verb and target. Table 2 is for informational purposes only; in case of a conflict between Table 2 and
 267 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 268 information in Table 2.

269 **Table 2 – Command Verb Requirements for CIM_ControlledBy**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.2.2.
start	Not supported	
stop	Not supported	

270 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 271 `reset`, `set`, `start`, and `stop`.

272 6.2.1 Ordering of Results

273 When results are returned for multiple instances of `CIM_ControlledBy`, implementations shall utilize the
 274 following algorithm to produce the natural (that is, default) ordering:

- 275 • Results for `CIM_ControlledBy` are unordered; therefore, no algorithm is defined.

276 6.2.2 Show

277 This section describes how to implement the `show` verb when applied to an instance of
 278 `CIM_ControlledBy`. Implementations shall support the use of the `show` verb with `CIM_ControlledBy`.

279 6.2.2.1 Show Command Form for Multiple Instances Target – CIM_PCIPort Reference

280 This command form is used to show many instances of `CIM_ControlledBy`. This command form
 281 corresponds to a `show` command issued against instances of `CIM_ControlledBy` where only one
 282 reference is specified and the reference is to an instance of `CIM_PCIPort`.

283 6.2.2.1.1 Command Form

284 `show <CIM_ControlledBy multiple instances>`

285 6.2.2.1.2 CIM Requirements

286 See `CIM_ControlledBy` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 287 properties.

288 6.2.2.1.3 Behavior Requirements

289 6.2.2.1.3.1 Preconditions

290 \$instance represents the instance of CIM_PCIPort which is referenced by CIM_ControlledBy.

291 6.2.2.1.3.2 Pseudo Code

```
292 &smShowAssociationInstances ( "CIM_ControlledBy", $instance.getObjectPath() );
293 &smEnd;
```

294 6.2.2.2 Show Command Form for a Single Instance – CIM_PCIDevice Reference

295 This command form is used to show a single instance of CIM_ControlledBy. This command form
296 corresponds to a `show` command issued against a single instance of CIM_ControlledBy where only one
297 reference is specified and the reference is to the instance of CIM_PCIDevice.

298 6.2.2.2.1 Command Form

```
299 show <CIM_ControlledBy single instance>
```

300 6.2.2.2.2 CIM Requirements

301 See CIM_ControlledBy in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
302 properties.

303 6.2.2.2.3 Behavior Requirements

304 6.2.2.2.3.1 Preconditions

305 \$instance represents the instance of CIM_PCIDevice which is referenced by CIM_ControlledBy.

306 6.2.2.2.3.2 Pseudo Code

```
307 &smShowAssociationInstances ( "CIM_ControlledBy", $instance.getObjectPath() );
308 &smEnd;
```

309 6.2.2.3 Show Command Form for a Single Instance Target – Both References

310 This command form is for the `show` verb applied to a single instance. This command form corresponds to
311 the `show` command issued against CIM_ControlledBy where both references are specified and therefore
312 the desired instance is unambiguously identified.

313 6.2.2.3.1 Command Form

```
314 show <CIM_ControlledBy single instance>
```

315 6.2.2.3.2 CIM Requirements

316 See CIM_ControlledBy in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
317 properties.

318 6.2.2.3.3 Behavior Requirements

319 6.2.2.3.3.1 Preconditions

320 \$instanceA represents the referenced instance of CIM_PCIDevice through the CIM_ControlledBy
321 association.

322 \$instanceB represents the instance of CIM_PCIPort which is referenced by CIM_ControlledBy.

323 **6.2.2.3.3.2 Pseudo Code**

```
324 &smShowAssociationInstance ( "CIM_ControlledBy", $instanceA.getObjectPath(),
325     $instanceB.getObjectPath() );
326 &smEnd;
```

327 **6.3 CIM_DeviceConnection**

328 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

329 Table 3 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 330 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 331 verb and target. Table 3 is for informational purposes only; in case of a conflict between Table 3 and
 332 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 333 information in Table 3.

334 **Table 3 – Command Verb Requirements for CIM_DeviceConnection**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.3.2.
start	Not supported	
stop	Not supported	

335 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 336 `reset`, `set`, `start`, and `stop`.

337 **6.3.1 Ordering of Results**

338 When results are returned for multiple instances of `CIM_DeviceConnection`, implementations shall utilize
 339 the following algorithm to produce the natural (that is, default) ordering:

- 340 • Results for `CIM_DeviceConnection` are unordered; therefore, no algorithm is defined.

341 **6.3.2 Show**

342 This section describes how to implement the `show` verb when applied to an instance of
 343 `CIM_DeviceConnection`. Implementations shall support the use of the `show` verb with
 344 `CIM_DeviceConnection`.

345 **6.3.2.1 Show Command Form for Multiple Instances Target – CIM_PCIPort Reference**

346 This command form is used to show many instances of `CIM_DeviceConnection`. This command form
 347 corresponds to a `show` command issued against instances of `CIM_DeviceConnection` where only one
 348 reference is specified and the reference is to an instance of `CIM_PCIPort`.

349 6.3.2.1.1 Command Form

```
350 show <CIM_DeviceConnection multiple instances>
```

351 6.3.2.1.2 CIM Requirements

352 See CIM_DeviceConnection in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
353 mandatory properties.

354 6.3.2.1.3 Behavior Requirements

355 6.3.2.1.3.1 Preconditions

356 \$instance represents the instance of CIM_PCIPort which is referenced by CIM_DeviceConnection.

357 6.3.2.1.3.2 Pseudo Code

```
358 &smShowAssociationInstances ( "CIM_DeviceConnection", $instance.getObjectPath() );  
359 &smEnd;
```

360 6.3.2.2 Show Command Form for a Single Instance – CIM_PCIPort Reference

361 This command form is used to show a single instance of CIM_DeviceConnection. This command form
362 corresponds to a show command issued against a single instance of CIM_DeviceConnection where only
363 one reference is specified and the reference is to the instance of CIM_PCIPort.

364 6.3.2.2.1 Command Form

```
365 show <CIM_DeviceConnection single instance>
```

366 6.3.2.2.2 CIM Requirements

367 See CIM_DeviceConnection in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
368 mandatory properties.

369 6.3.2.2.3 Behavior Requirements

370 6.3.2.2.3.1 Preconditions

371 \$instance represents the instance of CIM_PCIPort which is referenced by CIM_DeviceConnection.

372 6.3.2.2.3.2 Pseudo Code

```
373 &smShowAssociationInstances ( "CIM_DeviceConnection", $instance.getObjectPath() );  
374 &smEnd;
```

375 6.3.2.3 Show Command Form for a Single Instance Target – Both References

376 This command form is for the show verb applied to a single instance. This command form corresponds to
377 the show command issued against CIM_DeviceConnection where both references are specified and
378 therefore the desired instance is unambiguously identified.

379 6.3.2.3.1 Command Form

```
380 show <CIM_DeviceConnection single instance>
```

381 6.3.2.3.2 CIM Requirements

382 See CIM_DeviceConnection in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
383 mandatory properties.

384 **6.3.2.3.3 Behavior Requirements**385 **6.3.2.3.3.1 Preconditions**

386 \$instanceA represents the referenced instance of CIM_PCIPort through CIM_DeviceConnection
387 association.

388 \$instanceB represents the instance of CIM_PCIPort which is referenced by CIM_DeviceConnection.

389 **6.3.2.3.3.2 Pseudo Code**

```
390 &smShowAssociationInstance ( "CIM_DeviceConnection", $instanceA.getObjectPath(),
391     $instanceB.getObjectPath() );
392 &smEnd;
```

393 **6.4 CIM_HostedCollection**

394 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

395 Table 4 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
396 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
397 verb and target. Table 4 is for informational purposes only; in case of a conflict between Table 4 and
398 requirements detailed in the following sections, the text detailed in the following sections supersedes the
399 information in Table 4.

400

Table 4 – Command Verb Requirements for CIM_HostedCollection

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.4.2.
start	Not supported	
stop	Not supported	

401 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
402 `reset`, `set`, `start`, and `stop`.

403 **6.4.1 Ordering of Results**

404 When results are returned for multiple instances of CIM_HostedCollection, implementations shall utilize
405 the following algorithm to produce the natural (that is, default) ordering:

- 406
- Results for CIM_HostedCollection are unordered; therefore, no algorithm is defined.

407 6.4.2 Show

408 This section describes how to implement the `show` verb when applied to an instance of
409 `CIM_HostedCollection`. Implementations shall support the use of the `show` verb with
410 `CIM_HostedCollection`.

411 6.4.2.1 Show Command Form for Multiple Instances Target – `CIM_ComputerSystem` Reference

412 This command form is used to show many instances of `CIM_HostedCollection`. This command form
413 corresponds to a `show` command issued against instances of `CIM_HostedCollection` where only one
414 reference is specified and the reference is to an instance of `CIM_ComputerSystem`.

415 6.4.2.1.1 Command Form

```
416 show <CIM_HostedCollection multiple instances>
```

417 6.4.2.1.2 CIM Requirements

418 See `CIM_HostedCollection` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
419 mandatory properties.

420 6.4.2.1.3 Behavior Requirements

421 6.4.2.1.3.1 Preconditions

422 `$instance` represents the instance of `CIM_ComputerSystem` which is referenced by
423 `CIM_HostedCollection`.

424 6.4.2.1.3.2 Pseudo Code

```
425 &smShowAssociationInstances ( "CIM_HostedCollection", $instance.GetObjectPath() );  
426 &smEnd;
```

427 6.4.2.2 Show Command Form for a Single Instance – `CIM_PCIPortGroup` Reference

428 This command form is used to show a single instance of `CIM_HostedCollection`. This command form
429 corresponds to a `show` command issued against a single instance of `CIM_HostedCollection` where only
430 one reference is specified and the reference is to the instance of `CIM_PCIPortGroup`.

431 6.4.2.2.1 Command Form

```
432 show <CIM_HostedCollection single instance>
```

433 6.4.2.2.2 CIM Requirements

434 See `CIM_HostedCollection` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
435 mandatory properties.

436 6.4.2.2.3 Behavior Requirements

437 6.4.2.2.3.1 Preconditions

438 `$instance` represents the instance of `CIM_PCIPortGroup` which is referenced by `CIM_HostedCollection`.

439 6.4.2.2.3.2 Pseudo Code

```
440 &smShowAssociationInstances ( "CIM_HostedCollection", $instance.GetObjectPath() );  
441 &smEnd;
```

442 **6.4.2.3 Show Command Form for a Single Instance Target – Both References**

443 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 444 the `show` command issued against `CIM_HostedCollection` where both references are specified and
 445 therefore the desired instance is unambiguously identified.

446 **6.4.2.3.1 Command Form**

```
447 show <CIM_HostedCollection single instance>
```

448 **6.4.2.3.2 CIM Requirements**

449 See `CIM_HostedCollection` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
 450 mandatory properties.

451 **6.4.2.3.3 Behavior Requirements**

452 **6.4.2.3.3.1 Preconditions**

453 `$instanceA` represents the referenced instance of `CIM_ComputerSystem` through the
 454 `CIM_HostedCollection` association.

455 `$instanceB` represents the instance of `CIM_PCIPortGroup` which is referenced by `CIM_HostedCollection`.

456 **6.4.2.3.3.2 Pseudo Code**

```
457 &smShowAssociationInstance ( "CIM_HostedCollection", $instanceA.getObjectPath(),  
458     $instanceB.getObjectPath() );  
459 &smEnd;
```

460 **6.5 CIM_ConcretelDentity**

461 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

462 Table 5 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 463 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 464 verb and target. Table 5 is for informational purposes only; in case of a conflict between Table 5 and
 465 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 466 information in Table 5.

467 **Table 5 – Command Verb Requirements for CIM_ConcretelDentity**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.5.2.
start	Not supported	
stop	Not supported	

468 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 469 `reset`, `set`, `start`, and `stop`.

470 6.5.1 Ordering of Results

471 When results are returned for multiple instances of CIM_ConcreteIdentity, implementations shall utilize
472 the following algorithm to produce the natural (that is, default) ordering:

- 473 • Results for CIM_ConcreteIdentity are unordered; therefore, no algorithm is defined.

474 6.5.2 Show

475 This section describes how to implement the `show` verb when applied to an instance of
476 CIM_ConcreteIdentity. Implementations shall support the use of the `show` verb with
477 CIM_ConcreteIdentity.

478 6.5.2.1 Show Command Form for Multiple Instances Target – CIM_LogicalDevice Reference

479 This command form is used to show many instances of CIM_ConcreteIdentity. This command form
480 corresponds to a `show` command issued against instances of CIM_ConcreteIdentity where only one
481 reference is specified and the reference is to an instance of CIM_LogicalDevice.

482 6.5.2.1.1 Command Form

```
483 show <CIM_ConcreteIdentity multiple instances>
```

484 6.5.2.1.2 CIM Requirements

485 See CIM_ConcreteIdentity in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
486 mandatory properties.

487 6.5.2.1.3 Behavior Requirements

488 6.5.2.1.3.1 Preconditions

489 \$instance represents the instance of CIM_LogicalDevice which is referenced by CIM_ConcreteIdentity.

490 6.5.2.1.3.2 Pseudo Code

```
491 &smShowAssociationInstances ( "CIM_ConcreteIdentity", $instance.GetObjectPath() );  
492 &smEnd;
```

493 6.5.2.2 Show Command Form for a Single Instance – CIM_PCIDevice Reference

494 This command form is used to show a single instance of CIM_ConcreteIdentity. This command form
495 corresponds to a `show` command issued against a single instance of CIM_ConcreteIdentity where only
496 one reference is specified and the reference is to the instance of CIM_PCIDevice.

497 6.5.2.2.1 Command Form

```
498 show <CIM_ConcreteIdentity single instance>
```

499 6.5.2.2.2 CIM Requirements

500 See CIM_ConcreteIdentity in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
501 mandatory properties.

502 6.5.2.2.3 Behavior Requirements

503 6.5.2.2.3.1 Preconditions

504 \$instance represents the instance of CIM_PCIDevice which is referenced by CIM_ConcreteIdentity.

505 **6.5.2.2.3.2 Pseudo Code**

```
506 &smShowAssociationInstances ( "CIM_ConcreteIdentity", $instance.getObjectPath() );
507 &smEnd;
```

508 **6.5.2.3 Show Command Form for a Single Instance Target – Both References**

509 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 510 the `show` command issued against `CIM_ConcreteIdentity` where both references are specified and
 511 therefore the desired instance is unambiguously identified.

512 **6.5.2.3.1 Command Form**

```
513 show <CIM_ConcreteIdentity single instance>
```

514 **6.5.2.3.2 CIM Requirements**

515 See `CIM_ConcreteIdentity` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
 516 mandatory properties.

517 **6.5.2.3.3 Behavior Requirements**

518 **6.5.2.3.3.1 Preconditions**

519 `$instanceA` represents the referenced instance of `CIM_LogicalDevice` through `CIM_ConcreteIdentity`
 520 association.

521 `$instanceB` represents the instance of `CIM_PCIDevice` which is referenced by `CIM_ConcreteIdentity`.

522 **6.5.2.3.3.2 Pseudo Code**

```
523 &smShowAssociationInstance ( "CIM_ConcreteIdentity", $instanceA.getObjectPath(),
524     $instanceB.getObjectPath() );
525 &smEnd;
```

526 **6.6 CIM_MemberOfCollection**

527 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

528 Table 6 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 529 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 530 verb and target. Table 6 is for informational purposes only; in case of a conflict between Table 6 and
 531 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 532 information in Table 6.

533 **Table 6 – Command Verb Requirements for CIM_MemberOfCollection**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	

Command Verb	Requirement	Comments
show	Shall	See 6.6.2.
start	Not supported	
stop	Not supported	

534 No mapping is defined for the following verbs for the specified target: *create*, *delete*, *dump*, *load*,
535 *reset*, *set*, *start*, and *stop*.

536 6.6.1 Ordering of Results

537 When results are returned for multiple instances of *CIM_MemberOfCollection*, implementations shall
538 utilize the following algorithm to produce the natural (that is, default) ordering:

- 539 • Results for *CIM_MemberOfCollection* are unordered; therefore, no algorithm is defined.

540 6.6.2 Show

541 This section describes how to implement the *show* verb when applied to an instance of
542 *CIM_MemberOfCollection*. Implementations shall support the use of the *show* verb with
543 *CIM_MemberOfCollection*.

544 6.6.2.1 Show Command Form for Multiple Instances Target – *CIM_PCIPortGroup* Reference

545 This command form is used to show many instances of *CIM_MemberOfCollection*. This command form
546 corresponds to a *show* command issued against instances of *CIM_MemberOfCollection* where only one
547 reference is specified and the reference is to an instance of *CIM_PCIPortGroup*.

548 6.6.2.1.1 Command Form

```
549 show <CIM_MemberOfCollection multiple instances>
```

550 6.6.2.1.2 CIM Requirements

551 See *CIM_MemberOfCollection* in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
552 mandatory properties.

553 6.6.2.1.3 Behavior Requirements

554 6.6.2.1.3.1 Preconditions

555 \$instance represents the instance of *CIM_PCIPortGroup* which is referenced by
556 *CIM_MemberOfCollection*.

557 6.6.2.1.3.2 Pseudo Code

```
558 &smShowAssociationInstances ( "CIM_MemberOfCollection", $instance.getObjectPath() );  
559 &smEnd;
```

560 6.6.2.2 Show Command Form for a Single Instance – *CIM_PCIPort* Reference

561 This command form is used to show a single instance of *CIM_MemberOfCollection*. This command form
562 corresponds to a *show* command issued against a single instance of *CIM_MemberOfCollection* where
563 only one reference is specified and the reference is to the instance of *CIM_PCIPort*.

564 **6.6.2.2.1 Command Form**565

```
show <CIM_MemberOfCollection single instance>
```

566 **6.6.2.2.2 CIM Requirements**567 See CIM_MemberOfCollection in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
568 mandatory properties.569 **6.6.2.2.3 Behavior Requirements**570 **6.6.2.2.3.1 Preconditions**

571 \$instance represents the instance of CIM_PCIPort which is referenced by CIM_MemberOfCollection.

572 **6.6.2.2.3.2 Pseudo Code**573

```
&smShowAssociationInstances ( "CIM_MemberOfCollection", $instance.getObjectPath() );
```


574

```
&smEnd;
```

575 **6.6.2.3 Show Command Form for a Single Instance Target – Both References**576 This command form is for the `show` verb applied to a single instance. This command form corresponds to
577 the `show` command issued against CIM_MemberOfCollection where both references are specified and
578 therefore the desired instance is unambiguously identified.579 **6.6.2.3.1 Command Form**580

```
show <CIM_MemberOfCollection single instance>
```

581 **6.6.2.3.2 CIM Requirements**582 See CIM_MemberOfCollection in the “CIM Elements” section of the [PCI Device Profile](#) for the list of
583 mandatory properties.584 **6.6.2.3.3 Behavior Requirements**585 **6.6.2.3.3.1 Preconditions**586 \$instanceA represents the referenced instance of CIM_PCIPortGroup through the
587 CIM_MemberOfCollection association.

588 \$instanceB represents the instance of CIM_PCIPort which is referenced by CIM_MemberOfCollection.

589 **6.6.2.3.3.2 Pseudo Code**590

```
&smShowAssociationInstance ( "CIM_MemberOfCollection", $instanceA.getObjectPath(),
```


591

```
    $instanceB.getObjectPath() );
```


592

```
&smEnd;
```

593 **6.7 CIM_SystemDevice**594 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).595 Table 7 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
596 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
597 verb and target. Table 7 is for informational purposes only; in case of a conflict between Table 7 and
598 requirements detailed in the following sections, the text detailed in the following sections supersedes the
599 information in Table 7.

600

Table 7 – Command Verb Requirements for CIM_SystemDevice

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.7.2.
start	Not supported	
stop	Not supported	

601 No mapping is defined for the following verbs for the specified target: *create*, *delete*, *dump*, *load*,
602 *reset*, *set*, *start*, and *stop*.

603 6.7.1 Ordering of Results

604 When results are returned for multiple instances of *CIM_SystemDevice*, implementations shall utilize the
605 following algorithm to produce the natural (that is, default) ordering:

- 606 • Results for *CIM_SystemDevice* are unordered; therefore, no algorithm is defined.

607 6.7.2 Show

608 This section describes how to implement the *show* verb when applied to an instance of
609 *CIM_SystemDevice*. Implementations shall support the use of the *show* verb with *CIM_SystemDevice*.

610 6.7.2.1 Show Command Form for Multiple Instances Target – *CIM_ComputerSystem* Reference

611 This command form is used to show many instances of *CIM_SystemDevice*. This command form
612 corresponds to a *show* command issued against instances of *CIM_SystemDevice* where only one
613 reference is specified and the reference is to an instance of *CIM_ComputerSystem*.

614 6.7.2.1.1 Command Form

```
615 show <CIM_SystemDevice multiple instances>
```

616 6.7.2.1.2 CIM Requirements

617 See *CIM_SystemDevice* in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
618 properties.

619 6.7.2.1.3 Behavior Requirements

620 6.7.2.1.3.1 Preconditions

621 \$instance represents the instance of *CIM_ComputerSystem* which is referenced by *CIM_SystemDevice*.

622 6.7.2.1.3.2 Pseudo Code

```
623 &smShowAssociationInstances ( "CIM_SystemDevice", $instance.getObjectPath() );  
624 &smEnd;
```

625 **6.7.2.2 Show Command Form for a Single Instance – CIM_PCIDevice Reference**

626 This command form is used to show a single instance of CIM_SystemDevice. This command form
627 corresponds to a `show` command issued against a single instance of CIM_SystemDevice where only one
628 reference is specified and the reference is to the instance of CIM_PCIDevice.

629 **6.7.2.2.1 Command Form**

```
630 show <CIM_SystemDevice single instance>
```

631 **6.7.2.2.2 CIM Requirements**

632 See CIM_SystemDevice in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
633 properties.

634 **6.7.2.2.3 Behavior Requirements**

635 **6.7.2.2.3.1 Preconditions**

636 \$instance represents the instance of CIM_PCIDevice which is referenced by CIM_SystemDevice.

637 **6.7.2.2.3.2 Pseudo Code**

```
638 &smShowAssociationInstances ( "CIM_SystemDevice", $instance.getObjectPath() );  
639 &smEnd;
```

640 **6.7.2.3 Show Command Form for a Single Instance – CIM_PCIPort Reference**

641 This command form is used to show a single instance of CIM_SystemDevice. This command form
642 corresponds to a `show` command issued against a single instance of CIM_SystemDevice where only one
643 reference is specified and the reference is to the instance of CIM_PCIPort.

644 **6.7.2.3.1 Command Form**

```
645 show <CIM_SystemDevice single instance>
```

646 **6.7.2.3.2 CIM Requirements**

647 See CIM_SystemDevice in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
648 properties.

649 **6.7.2.3.3 Behavior Requirements**

650 **6.7.2.3.3.1 Preconditions**

651 \$instance represents the instance of CIM_PCIPort which is referenced by CIM_SystemDevice.

652 **6.7.2.3.3.2 Pseudo Code**

```
653 &smShowAssociationInstances ( "CIM_SystemDevice", $instance.getObjectPath() );  
654 &smEnd;
```

655 **6.7.2.4 Show Command Form for a Single Instance Target – CIM_ComputerSystem and 656 CIM_PCIDevice References**

657 This command form is for the `show` verb applied to a single instance. This command form corresponds to
658 the `show` command issued against CIM_SystemDevice where CIM_ComputerSystem and
659 CIM_PCIDevice references are specified and therefore the desired instance is unambiguously identified.

660 **6.7.2.4.1 Command Form**

661 `show <CIM_SystemDevice single instance>`

662 **6.7.2.4.2 CIM Requirements**

663 See CIM_SystemDevice in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
664 properties.

665 **6.7.2.4.3 Behavior Requirements**

666 **6.7.2.4.3.1 Preconditions**

667 \$instanceA represents the referenced instance of CIM_ComputerSystem through the CIM_SystemDevice
668 association.

669 \$instanceB represents the instance of CIM_PCIDevice which is referenced by CIM_SystemDevice.

670 **6.7.2.4.3.2 Pseudo Code**

```
671 &smShowAssociationInstance ( "CIM_SystemDevice", $instanceA.getObjectPath(),
672     $instanceB.getObjectPath() );
673 &smEnd;
```

674 **6.7.2.5 Show Command Form for a Single Instance Target – CIM_ComputerSystem and** 675 **CIM_PCIPort References**

676 This command form is for the show verb applied to a single instance. This command form corresponds to
677 the show command issued against CIM_SystemDevice where CIM_ComputerSystem and CIM_PCIPort
678 references are specified and therefore the desired instance is unambiguously identified.

679 **6.7.2.5.1 Command Form**

680 `show <CIM_SystemDevice single instance>`

681 **6.7.2.5.2 CIM Requirements**

682 See CIM_SystemDevice in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
683 properties.

684 **6.7.2.5.3 Behavior Requirements**

685 **6.7.2.5.3.1 Preconditions**

686 \$instanceA represents the referenced instance of CIM_ComputerSystem through CIM_SystemDevice
687 association.

688 \$instanceB represents the instance of CIM_PCIPort which is referenced by CIM_SystemDevice.

689 **6.7.2.5.3.2 Pseudo Code**

```
690 &smShowAssociationInstance ( "CIM_SystemDevice", $instanceA.getObjectPath(),
691     $instanceB.getObjectPath() );
692 &smEnd;
```

693 **6.8 CIM_PCIPort**

694 The cd, exit, help, and version verbs shall be supported as described in [DSP0216](#).

695 Table 8 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 696 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 697 verb and target. Table 8 is for informational purposes only; in case of a conflict between Table 8 and
 698 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 699 information in Table 8.

700 **Table 8 – Command Verb Requirements for CIM_PCIPort**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.8.2.
start	Not supported	
stop	Not supported	

701 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,
 702 reset, set, start, and stop.

703 6.8.1 Ordering of Results

704 When results are returned for multiple instances of CIM_PCIPort, implementations shall utilize the
 705 following algorithm to produce the natural (that is, default) ordering:

- 706 • Results for CIM_PCIPort are unordered; therefore, no algorithm is defined.

707 6.8.2 Show

708 This section describes how to implement the `show` verb when applied to an instance of CIM_PCIPort.
 709 Implementations shall support the use of the `show` verb with CIM_PCIPort.

710 6.8.2.1 Show Command Form for Multiple Instances Target

711 This command form is used to show many instances of CIM_PCIPort.

712 6.8.2.1.1 Command Form

```
713 show <CIM_PCIPort multiple instances>
```

714 6.8.2.1.2 CIM Requirements

715 See CIM_PCIPort in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 716 properties.

717 6.8.2.1.3 Behavior Requirements

718 6.8.2.1.3.1 Preconditions

719 \$containerInstance represents the instance of CIM_ComputerSystem which represents the container
720 system and is associated to the targeted instances of CIM_PCIPort through the CIM_SystemDevice
721 association.

722 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

723 6.8.2.1.3.2 Pseudo Code

```
724 #propertylist[] = NULL;
725 if ( false == #all )
726 {
727     #propertylist[] = <array of mandatory non-key property names (see CIM
728     Requirements)>;
729 }
730 &smShowInstances ( "CIM_PCIPort", "CIM_SystemDevice",
731     $containerInstance.getObjectPath(), #propertylist[] );
732 &smEnd;
```

733 6.8.2.2 Show Command Form for a Single Instance Target

734 This command form is used to show a single instance of CIM_PCIPort.

735 6.8.2.2.1 Command Form

```
736 show <CIM_PCIPort single instance>
```

737 6.8.2.2.2 CIM Requirements

738 See CIM_PCIPort in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
739 properties.

740 6.8.2.2.3 Behavior Requirements

741 6.8.2.2.3.1 Preconditions

742 \$instance represents the targeted instance of CIM_PCIPort.

```
743 $instance=<CIM_PCIPort single instance>;
```

744 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

745 6.8.2.2.3.2 Pseudo Code

```
746 #propertylist[] = NULL;
747 if ( false == #all )
748 {
749     #propertylist[] = <array of mandatory non-key property names (see CIM
750     Requirements)>;
751 }
752 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
753 &smEnd;
```

754 **6.9 CIM_PCIPortGroup**755 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

756 Table 9 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 757 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 758 verb and target. Table 9 is for informational purposes only; in case of a conflict between Table 9 and
 759 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 760 information in Table 9.

761 **Table 9 – Command Verb Requirements for CIM_PCIPortGroup**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not Support	
show	Shall	See 6.9.2.
start	Not supported	
stop	Not supported	

762 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`, `reset`,
 763 `set`, `start`, and `stop`.

764 **6.9.1 Ordering of Results**

765 When results are returned for multiple instances of `CIM_PCIPortGroup`, implementations shall utilize the
 766 following algorithm to produce the natural (that is, default) ordering:

- 767 • Results for `CIM_PCIPortGroup` are unordered; therefore, no algorithm is defined.

768 **6.9.2 Show**

769 This section describes how to implement the `show` verb when applied to an instance of
 770 `CIM_PCIPortGroup`. Implementations shall support the use of the `show` verb with `CIM_PCIPortGroup`.

771 **6.9.2.1 Show Command Form for Multiple Instances Target**

772 This command form is used to show many instances of `CIM_PCIPortGroup`.

773 **6.9.2.1.1 Command Form**

774 `show <CIM_PCIPortGroup multiple instances>`

775 **6.9.2.1.2 CIM Requirements**

776 See `CIM_PCIPortGroup` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 777 properties.

778 6.9.2.1.3 Behavior Requirements

779 6.9.2.1.3.1 Preconditions

780 \$containerInstance represents the instance of CIM_ComputerSystem which represents the container
781 system and is associated to the targeted instances of CIM_PCIPortGroup through the
782 CIM_HostedCollection association.

783 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

784 6.9.2.1.3.2 Pseudo Code

```
785 #propertylist[] = NULL;
786 if ( false == #all )
787     {
788         #propertylist[] = <array of mandatory non-key property names (see CIM
789             Requirements)>;
790     }
791 &smShowInstances ( "CIM_PCIPortGroup", "CIM_HostedCollection",
792     $containerInstance.getObjectPath(), #propertylist[] );
793 &smEnd;
```

794 6.9.2.2 Show Command Form for a Single Instance Target

795 This command form is used to show a single instance of CIM_PCIPortGroup.

796 6.9.2.2.1 Command Form

```
797 show <CIM_PCIPortGroup single instance>
```

798 6.9.2.2.2 CIM Requirements

799 See CIM_PCIPortGroup in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
800 properties.

801 6.9.2.2.3 Behavior Requirements

802 6.9.2.2.3.1 Preconditions

803 \$instance represents the targeted instance of CIM_PCIPortGroup.

```
804 $instance=<CIM_PCIPortGroup single instance>;
```

805 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

806 6.9.2.2.3.2 Pseudo Code

```
807 #propertylist[] = NULL;
808 if ( false == #all )
809     {
810         #propertylist[] = <array of mandatory non-key property names (see CIM
811             Requirements)>;
812     }
813 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
814 &smEnd;
```

815 **6.10 CIM_PCIDevice**

816 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

817 Table 10 lists each SM CLP verb, the required level of support for the verb in conjunction with instances
 818 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 819 verb and target. Table 10 is for informational purposes only; in case of a conflict between Table 10 and
 820 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 821 information in Table 10.

822 **Table 10 – Command Verb Requirements for CIM_PCIDevice**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	May	See 6.10.2.
set	May	See 6.10.3.
show	Shall	See 6.10.4.
start	May	See 6.10.5.
stop	May	See 6.10.6.

823 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

824 **6.10.1 Ordering of Results**

825 When results are returned for multiple instances of CIM_PCIDevice, implementations shall utilize the
 826 following algorithm to produce the natural (that is, default) ordering:

- 827 • Results for CIM_PCIDevice are unordered; therefore, no algorithm is defined.

828 **6.10.2 Reset**

829 **6.10.2.1 General Usage of Set for a Single Property**

830 This section describes how to implement the `reset` verb when applied to an instance of CIM_PCIDevice.
 831 Implementations may support the use of the `reset` verb with CIM_PCIDevice.

832 **6.10.2.1.1 Command Form**

833 `reset <CIM_PCIDevice single instance>`

834 **6.10.2.1.2 CIM Requirements**

```

835 uint16 EnabledState;
836 uint16 RequestedState;
837 uint32 CIM_PCIDevice.RequestStateChange (
838     [IN] uint16 RequestedState,
839     [OUT] REF CIM_ConcreteJob Job,
840     [IN] datetime TimeoutPeriod );
    
```

841 6.10.2.1.3 Behavior Requirements

842 6.10.2.1.3.1 Preconditions

843 \$instance represents the targeted instance of CIM_PCIDevice.

```
844 $instance=<CIM_PCIDevice single instance>;
```

845 6.10.2.1.3.2 Pseudo Code

```
846 &smResetRSC ( $instance.GetObjectPath() );
847 &smEnd;
```

848 6.10.3 Set

849 This section describes how to implement the `set` verb when it is applied to an instance of
850 CIM_PCIDevice. Implementations may support the use of the `set` verb with CIM_PCIDevice.

851 The `set` verb is used to modify descriptive properties of the CIM_PCIDevice instance.

852 6.10.3.1 General Usage of Set for a Single Property

853 This command form corresponds to the general usage of the `set` verb to modify a single property of a
854 target instance. This is the most common case.

855 The requirement for supporting modification of a property using this command form shall be equivalent to
856 the requirement for supporting modification of the property using the ModifyInstance operation as defined
857 in the [PCI Device Profile](#).

858 6.10.3.1.1 Command Form

```
859 set <CIM_PCIDevice single instance> <propertyname>=<propertyvalue>
```

860 6.10.3.1.2 CIM Requirements

861 See CIM_PCIDevice in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
862 properties.

863 6.10.3.1.3 Behavior Requirements

864 6.10.3.1.3.1 Preconditions

```
865 $instance=<CIM_PCIDevice single instance>;
```

866 6.10.3.1.3.2 Pseudo Code

```
867 #propertyName[] = {<propertyname>};
868 #propertyValues[] = {<propertyvalue>};
869 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
870 &smEnd;
```

871 6.10.3.2 General Usage of Set for Multiple Properties

872 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
873 target instance where there is not an explicit relationship between the properties. This is the most
874 common case.

875 The requirement for supporting modification of a property using this command form shall be equivalent to
 876 the requirement for supporting modification of the property using the ModifyInstance operation as defined
 877 in the [PCI Device Profile](#).

878 6.10.3.2.1 Command Form

```
879 set <CIM_PCIDevice single instance> <propertyname1>=<propertyvalue1>  
880 <propertynamen>=<propertyvaluen>
```

881 6.10.3.2.2 CIM Requirements

882 See CIM_PCIDevice in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 883 properties.

884 6.10.3.2.3 Behavior Requirements

885 6.10.3.2.3.1 Preconditions

```
886 $instance=<CIM_PCIDevice single instance>;
```

887 6.10.3.2.3.2 Pseudo Code

```
888 #propertyNames[] = {<propertyname>;}  
889 for #i < n  
890 {  
891     #propertyNames[#i] = <propertyname#i>  
892     #propertyValues[#i] = <propertyvalue#i>  
893 }  
894 &smSetInstance ( $instance, #propertyNames[], #propertyValues[] );  
895 &smEnd;
```

896 6.10.3.3 Set RequestedState to “Enabled”

897 This section describes how to change the state of the PCI device represented by CIM_PCIDevice to
 898 “Enabled”.

899 6.10.3.3.1 Command Form

```
900 set <CIM_PCIDevice single instance> RequestedState="Enabled"
```

901 6.10.3.3.2 CIM Requirements

902 See CIM_PCIDevice in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 903 properties.

904 6.10.3.3.3 Behavior Requirements

905 6.10.3.3.3.1 Preconditions

906 \$instance represents the targeted instance of CIM_PCIDevice.

```
907 $instance=<CIM_PCIDevice single instance>;
```

908 6.10.3.3.3.2 Pseudo Code

```
909 // "Enabled" is valuemap 2  
910 &smRequestStateChange ( $instance.getObjectPath(), 2 );  
911 &smEnd;
```


912 **6.10.4 Show**

913 This section describes how to implement the `show` verb when applied to an instance of `CIM_PCIDevice`.
 914 Implementations shall support the use of the `show` verb with `CIM_PCIDevice`.

915 **6.10.4.1 Show Command Form for Multiple Instances Target**

916 This command form is used to show many instances of `CIM_PCIDevice`.

917 **6.10.4.1.1 Command Form**

```
918 show <CIM_PCIDevice multiple instances>
```

919 **6.10.4.1.2 CIM Requirements**

920 See `CIM_PCIDevice` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 921 properties.

922 **6.10.4.1.3 Behavior Requirements**

923 **6.10.4.1.3.1 Preconditions**

924 `$containerInstance` represents the instance of `CIM_ComputerSystem` which represents the container
 925 system and is associated to the targeted instances of `CIM_PCIDevice` through the `CIM_SystemDevice`
 926 association.

927 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

928 **6.10.4.1.3.2 Pseudo Code**

```
929 #propertylist[] = NULL;  

  930 if ( false == #all )  

  931 {  

  932     #propertylist[] = <array of mandatory non-key property names (see CIM  

  933         Requirements)>;  

  934 }  

  935 &smShowInstances ( "CIM_PCIDevice", "CIM_SystemDevice",  

  936     $containerInstance.GetObjectPath(), #propertylist[] );  

  937 &smEnd;
```

938 **6.10.4.2 Show Command Form for a Single Instance Target**

939 This command form is used to show a single instance of `CIM_PCIDevice`.

940 **6.10.4.2.1 Command Form**

```
941 show <CIM_PCIDevice single instance>
```

942 **6.10.4.2.2 CIM Requirements**

943 See `CIM_PCIDevice` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 944 properties.

945 **6.10.4.2.3 Behavior Requirements**

946 **6.10.4.2.3.1 Preconditions**

947 `$instance` represents the targeted instance of `CIM_PCIDevice`.

```
948 $instance=<CIM_PCIDevice single instance>;
```

949 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

950 6.10.4.2.3.2 Pseudo Code

```

951 #propertylist[] = NULL;
952 if ( false == #all )
953 {
954     #propertylist[] = <array of mandatory non-key property names (see CIM
955         Requirements)>;
956 }
957 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
958 &smEnd;

```

959 6.10.5 Start

960 6.10.5.1 General Usage of Start for a Single Property

961 This section describes how to implement the `start` verb when applied to an instance of `CIM_PCIDevice`.
 962 Implementations may support the use of the `start` verb with `CIM_PCIDevice`.

963 6.10.5.1.1 Command Form

```
964 start <CIM_PCIDevice single instance>
```

965 6.10.5.1.2 CIM Requirements

```

966 uint16 EnabledState;
967 uint16 RequestedState;
968 uint32 CIM_PCIDevice.RequestStateChange (
969     [IN] uint16 RequestedState,
970     [OUT] REF CIM_ConcreteJob Job,
971     [IN] datetime TimeoutPeriod );

```

972 6.10.5.1.3 Behavior Requirements

973 6.10.5.1.3.1 Preconditions

974 `$instance` represents the targeted instance of `CIM_PCIDevice`.

```
975 $instance=<CIM_PCIDevice single instance>;
```

976 6.10.5.1.3.2 Pseudo Code

```

977 &smStartRSC ( $instance.getObjectPath() );
978 &smEnd;

```

979 6.10.6 Stop

980 6.10.6.1 General Usage of Stop for a Single Property

981 This section describes how to implement the `stop` verb when applied to an instance of `CIM_PCIDevice`.
 982 Implementations may support the use of the `stop` verb with `CIM_PCIDevice`.

983 6.10.6.1.1 Command Form

```
984 stop <CIM_PCIDevice single instance>
```

985 **6.10.6.1.2 CIM Requirements**

```

986 uint16 EnabledState;
987 uint16 RequestedState;
988 uint32 CIM_PCIDevice.RequestStateChange (
989     [IN] uint16 RequestedState,
990     [OUT] REF CIM_ConcreteJob Job,
991     [IN] datetime TimeoutPeriod );
    
```

992 **6.10.6.1.3 Behavior Requirements**

993 **6.10.6.1.3.1 Preconditions**

994 \$instance represents the targeted instance of CIM_PCIDevice.

```

995 $instance=<CIM_PCIDevice single instance>;
    
```

996 **6.10.6.1.3.2 Pseudo Code**

```

997 &smStopRSC ( $instance.getObjectPath() );
998 &smEnd;
    
```

999 **6.11 CIM_PCIBridge**

1000 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

1001 Table 11 lists each SM CLP verb, the required level of support for the verb in conjunction with instances
 1002 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 1003 verb and target. Table 11 is for informational purposes only; in case of a conflict between Table 11 and
 1004 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 1005 information in Table 11.

1006 **Table 11 – Command Verb Requirements for CIM_PCIBridge**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	May	See 6.11.2.
set	May	See 6.11.3.
show	Shall	See 6.11.4.
start	May	See 6.11.5.
stop	May	See 6.11.6.

1007 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

1008 **6.11.1 Ordering of Results**

1009 When results are returned for multiple instances of CIM_PCIBridge, implementations shall utilize the
 1010 following algorithm to produce the natural (that is, default) ordering:

- 1011 • Results for CIM_PCIBridge are unordered; therefore, no algorithm is defined.

1012 **6.11.2 Reset**1013 **6.11.2.1 General Usage of Reset for a Single Property**

1014 This section describes how to implement the `reset` verb when applied to an instance of `CIM_PCIBridge`.
 1015 Implementations may support the use of the `reset` verb with `CIM_PCIBridge`.

1016 **6.11.2.1.1 Command Form**

```
1017 reset <CIM_PCIBridge single instance>
```

1018 **6.11.2.1.2 CIM Requirements**

```
1019 uint16 EnabledState;
1020 uint16 RequestedState;
1021 uint32 CIM_PCIBridge.RequestStateChange (
1022     [IN] uint16 RequestedState,
1023     [OUT] REF CIM_ConcreteJob Job,
1024     [IN] datetime TimeoutPeriod );
```

1025 **6.11.2.1.3 Behavior Requirements**1026 **6.11.2.1.3.1 Preconditions**

1027 `$instance` represents the targeted instance of `CIM_PCIBridge`.

```
1028 $instance=<CIM_PCIBridge single instance>;
```

1029 **6.11.2.1.3.2 Pseudo Code**

```
1030 &smResetRSC ( $instance.getObjectPath() );
1031 &smEnd;
```

1032 **6.11.3 Set**

1033 This section describes how to implement the `set` verb when it is applied to an instance of
 1034 `CIM_PCIBridge`. Implementations may support the use of the `set` verb with `CIM_PCIBridge`.

1035 The `set` verb is used to modify descriptive properties of the `CIM_PCIBridge` instance.

1036 **6.11.3.1 General Usage of Set for a Single Property**

1037 This command form corresponds to the general usage of the `set` verb to modify a single property of a
 1038 target instance. This is the most common case.

1039 The requirement for supporting modification of a property using this command form shall be equivalent to
 1040 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
 1041 in the [PCI Device Profile](#).

1042 **6.11.3.1.1 Command Form**

```
1043 set <CIM_PCIBridge single instance> <propertyname>=<propertyvalue>
```

1044 **6.11.3.1.2 CIM Requirements**

1045 See `CIM_PCIBridge` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 1046 properties.

1047 6.11.3.1.3 Behavior Requirements

1048 6.11.3.1.3.1 Preconditions

```
1049 $instance=<CIM_PCIBridge single instance>
```

1050 6.11.3.1.3.2 Pseudo Code

```
1051 #propertyName[] = {<propertyname>};
1052 #propertyValues[] = {<propertyvalue>};
1053 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1054 &smEnd;
```

1055 6.11.3.2 General Usage of Set for Multiple Properties

1056 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
1057 target instance where there is not an explicit relationship between the properties. This is the most
1058 common case.

1059 The requirement for supporting modification of a property using this command form shall be equivalent to
1060 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
1061 in the [PCI Device Profile](#).

1062 6.11.3.2.1 Command Form

```
1063 set <CIM_PCIBridge single instance> <propertyname1>=<propertyvalue1>
1064 <propertynamen>=<propertyvaluen>
```

1065 6.11.3.2.2 CIM Requirements

1066 See `CIM_PCIBridge` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
1067 properties.

1068 6.11.3.2.3 Behavior Requirements

1069 6.11.3.2.3.1 Preconditions

```
1070 $instance=<CIM_PCIBridge single instance>;
```

1071 6.11.3.2.3.2 Pseudo Code

```
1072 #propertyName[] = {<propertyname>};
1073 for #i < n
1074 {
1075     #propertyName[#i] = <propertyname#i>
1076     #propertyValues[#i] = <propertyvalue#i>
1077 }
1078 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1079 &smEnd;
```

1080 6.11.3.3 Set RequestedState to “Enabled”

1081 This section describes how to change the state of the PCI device represented by `CIM_PCIBridge` to
1082 “Enabled”.

1083 6.11.3.3.1 Command Form

```
1084 set <CIM_PCIBridge single instance> RequestedState="Enabled"
```

1085 **6.11.3.3.2 CIM Requirements**

1086 See CIM_PCIBridge in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
1087 properties.

1088 **6.11.3.3.3 Behavior Requirements**1089 **6.11.3.3.3.1 Preconditions**

1090 \$instance represents the targeted instance of CIM_PCIBridge.

```
1091 $instance=<CIM_PCIBridge single instance>;
```

1092 **6.11.3.3.3.2 Pseudo Code**

```
1093 //”Enabled” is valuemap 2
1094 &smRequestStateChange ( $instance.getObjectPath(), 2 );
1095 &smEnd;
```

1096 **6.11.4 Show**

1097 This section describes how to implement the `show` verb when applied to an instance of CIM_PCIBridge.
1098 Implementations shall support the use of the `show` verb with CIM_PCIBridge.

1099 **6.11.4.1 Show Command Form for Multiple Instances Target**

1100 This command form is used to show many instances of CIM_PCIBridge.

1101 **6.11.4.1.1 Command Form**

```
1102 show <CIM_PCIBridge multiple instances>
```

1103 **6.11.4.1.2 CIM Requirements**

1104 See CIM_PCIBridge in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
1105 properties.

1106 **6.11.4.1.3 Behavior Requirements**1107 **6.11.4.1.3.1 Preconditions**

1108 \$containerInstance represents the instance of CIM_ComputerSystem which represents the container
1109 system and is associated to the targeted instances of CIM_PCIBridge through the CIM_SystemDevice
1110 association.

1111 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1112 **6.11.4.1.3.2 Pseudo Code**

```
1113 #propertylist[] = NULL;
1114 if ( false == #all )
1115 {
1116     #propertylist[] = <array of mandatory non-key property names (see CIM
1117         Requirements)>;
1118 }
1119 &smShowInstances ( "CIM_PCIBridge", "CIM_SystemDevice",
1120     $containerInstance.getObjectPath(), #propertylist[] );
1121 &smEnd;
```

1122 6.11.4.2 Show Command Form for a Single Instance Target

1123 This command form is used to show a single instance of CIM_PCIBridge.

1124 6.11.4.2.1 Command Form

```
1125 show <CIM_PCIBridge single instance>
```

1126 6.11.4.2.2 CIM Requirements

1127 See CIM_PCIBridge in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
1128 properties.

1129 6.11.4.2.3 Behavior Requirements

1130 6.11.4.2.3.1 Preconditions

1131 \$instance represents the targeted instance of CIM_PCIBridge.

```
1132 $instance=<CIM_PCIBridge single instance>;
```

1133 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1134 6.11.4.2.3.2 Pseudo Code

```
1135 #propertylist[] = NULL;
1136 if ( false == #all )
1137 {
1138     #propertylist[] = <array of mandatory non-key property names (see CIM
1139     Requirements)>;
1140 }
1141 &smShowInstance ( $instance.getObjectPath(), #propertylist[] );
1142 &smEnd;
```

1143 6.11.5 Start

1144 6.11.5.1 General Usage of Start for a Single Property

1145 This section describes how to implement the `start` verb when applied to an instance of CIM_PCIBridge.
1146 Implementations may support the use of the `start` verb with CIM_PCIBridge.

1147 6.11.5.1.1 Command Form

```
1148 start <CIM_PCIBridge single instance>
```

1149 6.11.5.1.2 CIM Requirements

```
1150 uint16 EnabledState;
1151 uint16 RequestedState;
1152 uint32 CIM_PCIBridge.RequestStateChange (
1153     [IN] uint16 RequestedState,
1154     [OUT] REF CIM_ConcreteJob Job,
1155     [IN] datetime TimeoutPeriod );
```

1156 6.11.5.1.3 Behavior Requirements

1157 6.11.5.1.3.1 Preconditions

1158 \$instance represents the targeted instance of CIM_PCIBridge.

```
1159 $instance=<CIM_PCIBridge single instance>;
```

1160 **6.11.5.1.3.2 Pseudo Code**

```
1161 &smStartRSC ( $instance.GetObjectPath() );
1162 &smEnd;
```

1163 **6.11.6 Stop**

1164 **6.11.6.1 General Usage of Stop for a Single Property**

1165 This section describes how to implement the `stop` verb when applied to an instance of `CIM_PCIBridge`.
 1166 Implementations may support the use of the `stop` verb with `CIM_PCIBridge`.

1167 **6.11.6.1.1 Command Form**

```
1168 stop <CIM_PCIBridge single instance>
```

1169 **6.11.6.1.2 CIM Requirements**

```
1170 uint16 EnabledState;
1171 uint16 RequestedState;
1172 uint32 CIM_PCIBridge.RequestStateChange (
1173     [IN] uint16 RequestedState,
1174     [OUT] REF CIM_ConcreteJob Job,
1175     [IN] datetime TimeoutPeriod );
```

1176 **6.11.6.1.3 Behavior Requirements**

1177 **6.11.6.1.3.1 Preconditions**

1178 `$instance` represents the targeted instance of `CIM_PCIBridge`.

```
1179 $instance=<CIM_PCIBridge single instance>;
```

1180 **6.11.6.1.3.2 Pseudo Code**

```
1181 &smStopRSC ( $instance.GetObjectPath() );
1182 &smEnd;
```

1183 **6.12 CIM_PClSwitch**

1184 The `cd`, `exit`, `help`, and `version` verbs shall be supported as described in [DSP0216](#).

1185 Table 12 lists each SM CLP verb, the required level of support for the verb in conjunction with instances
 1186 of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 1187 verb and target. Table 12 is for informational purposes only; in case of a conflict between Table 12 and
 1188 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 1189 information in Table 12.

1190 **Table 12 – Command Verb Requirements for CIM_PClSwitch**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	

Command Verb	Requirement	Comments
reset	May	See 6.12.2.
set	May	See 6.12.3.
show	Shall	See 6.12.4.
start	May	See 6.12.5.
stop	May	See 6.12.6.

1191 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

1192 6.12.1 Ordering of Results

1193 When results are returned for multiple instances of `CIM_PCISwitch`, implementations shall utilize the
1194 following algorithm to produce the natural (that is, default) ordering:

- 1195 • Results for `CIM_PCISwitch` are unordered; therefore, no algorithm is defined.

1196 6.12.2 Reset

1197 6.12.2.1 General Usage of Reset for a Single Property

1198 This section describes how to implement the `reset` verb when applied to an instance of
1199 `CIM_PCISwitch`. Implementations may support the use of the `reset` verb with `CIM_PCISwitch`.

1200 6.12.2.1.1 Command Form

```
1201 reset <CIM_PCISwitch single instance>
```

1202 6.12.2.1.2 CIM Requirements

```
1203 uint16 EnabledState;
1204 uint16 RequestedState;
1205 uint32 CIM_PCISwitch.RequestStateChange (
1206     [IN] uint16 RequestedState,
1207     [OUT] REF CIM_ConcreteJob Job,
1208     [IN] datetime TimeoutPeriod );
```

1209 6.12.2.1.3 Behavior Requirements

1210 6.12.2.1.3.1 Preconditions

1211 `$instance` represents the targeted instance of `CIM_PCISwitch`.

```
1212 $instance=<CIM_PCISwitch single instance>;
```

1213 6.12.2.1.3.2 Pseudo Code

```
1214 &smResetRSC ( $instance.GetObjectPath() );
1215 &smEnd;
```

1216 6.12.3 Set

1217 This section describes how to implement the `set` verb when it is applied to an instance of
1218 `CIM_PCISwitch`. Implementations may support the use of the `set` verb with `CIM_PCISwitch`.

1219 The `set` verb is used to modify descriptive properties of the `CIM_PCISwitch` instance.

1220 6.12.3.1 General Usage of Set for a Single Property

1221 This command form corresponds to the general usage of the `set` verb to modify a single property of a
1222 target instance. This is the most common case.

1223 The requirement for supporting modification of a property using this command form shall be equivalent to
1224 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
1225 in the [PCI Device Profile](#).

1226 6.12.3.1.1 Command Form

```
1227 set <CIM_PCISwitch single instance> <propertyname>=<propertyvalue>
```

1228 6.12.3.1.2 CIM Requirements

1229 See `CIM_PCISwitch` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
1230 properties.

1231 6.12.3.1.3 Behavior Requirements

1232 6.12.3.1.3.1 Preconditions

```
1233 $instance=<CIM_PCISwitch single instance>;
```

1234 6.12.3.1.3.2 Pseudo Code

```
1235 #propertyName[] = {<propertyname>};  
1236 #propertyValues[] = {<propertyvalue>};  
1237 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );  
1238 &smEnd;
```

1239 6.12.3.2 General Usage of Set for Multiple Properties

1240 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
1241 target instance where there is not an explicit relationship between the properties. This is the most
1242 common case.

1243 The requirement for supporting modification of a property using this command form shall be equivalent to
1244 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
1245 in the [PCI Device Profile](#).

1246 6.12.3.2.1 Command Form

```
1247 set <CIM_PCISwitch single instance> <propertyname1>=<propertyvalue1>  
1248 <propertynamen>=<propertyvaluen>
```

1249 6.12.3.2.2 CIM Requirements

1250 See `CIM_PCISwitch` in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
1251 properties.

1252 6.12.3.2.3 Behavior Requirements

1253 6.12.3.2.3.1 Preconditions

```
1254 $instance=<CIM_PCISwitch single instance>;
```

1255 6.12.3.2.3.2 Pseudo Code

```

1256 #propertyName[] = {<propertyname>};
1257 for #i < n
1258     {
1259         #propertyName[#i] = <propertyname#i>
1260         #propertyValues[#i] = <propertyvalue#i>
1261     }
1262 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
1263 &smEnd;
```

1264 6.12.3.3 Set RequestedState to “Enabled”

1265 This section describes how to change the state of the PCI device represented by CIM_PCIESwitch to
 1266 “Enabled”.

1267 6.12.3.3.1 Command Form

```
1268 set <CIM_PCIESwitch single instance> RequestedState="Enabled"
```

1269 6.12.3.3.2 CIM Requirements

1270 See CIM_PCIESwitch in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 1271 properties.

1272 6.12.3.3.3 Behavior Requirements

1273 6.12.3.3.3.1 Preconditions

1274 \$instance represents the targeted instance of CIM_PCIESwitch.

```
1275 $instance=<CIM_PCIESwitch single instance>;
```

1276 6.12.3.3.3.2 Pseudo Code

```

1277 //“Enabled” is valuemap 2
1278 &smRequestStateChange ( $instance.getObjectPath(), 2 );
1279 &smEnd;
```

1280 6.12.4 Show

1281 This section describes how to implement the `show` verb when applied to an instance of CIM_PCIESwitch.
 1282 Implementations shall support the use of the `show` verb with CIM_PCIESwitch.

1283 6.12.4.1 Show Command Form for Multiple Instances Target

1284 This command form is used to show many instances of CIM_PCIESwitch.

1285 6.12.4.1.1 Command Form

```
1286 show <CIM_PCIESwitch multiple instances>
```

1287 6.12.4.1.2 CIM Requirements

1288 See CIM_PCIESwitch in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 1289 properties.

1290 **6.12.4.1.3 Behavior Requirements**1291 **6.12.4.1.3.1 Preconditions**

1292 \$containerInstance represents the instance of CIM_ComputerSystem which represents the container
 1293 system and is associated to the targeted instances of CIM_PCIeSwitch through the CIM_SystemDevice
 1294 association.

1295 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1296 **6.12.4.1.3.2 Pseudo Code**

```
1297 #propertylist[] = NULL;
1298 if ( false == #all)
1299     {
1300         #propertylist[] = <array of mandatory non-key property names (see CIM
1301             Requirements)>;
1302     }
1303 &smShowInstances ( "CIM_PCIeSwitch", "CIM_SystemDevice",
1304     $containerInstance.getObjectPath(), #propertylist[] );
1305 &smEnd;
```

1306 **6.12.4.2 Show Command Form for a Single Instance Target**

1307 This command form is used to show a single instance of CIM_PCIeSwitch.

1308 **6.12.4.2.1 Command Form**

```
1309 show <CIM_PCIeSwitch single instance>
```

1310 **6.12.4.2.2 CIM Requirements**

1311 See CIM_PCIeSwitch in the “CIM Elements” section of the [PCI Device Profile](#) for the list of mandatory
 1312 properties.

1313 **6.12.4.2.3 Behavior Requirements**1314 **6.12.4.2.3.1 Preconditions**

1315 \$instance represents the targeted instance of CIM_PCIeSwitch.

```
1316 $instance=<CIM_PCIeSwitch single instance>;
```

1317 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1318 **6.12.4.2.3.2 Pseudo Code**

```
1319 #propertylist[] = NULL;
1320 if ( false == #all)
1321     {
1322         #propertylist[] = <array of mandatory non-key property names (see CIM
1323             Requirements)>;
1324     }
1325 &smShowInstance( $instance.getObjectPath(), #propertylist[] );
1326 &smEnd;
```

1327 **6.12.5 Start**1328 **6.12.5.1 General Usage of Start for a Single Property**

1329 This section describes how to implement the `start` verb when applied to an instance of
 1330 CIM_PCIESwitch. Implementations may support the use of the `start` verb with CIM_PCIESwitch.

1331 **6.12.5.1.1 Command Form**

```
1332 start <CIM_PCIESwitch single instance>
```

1333 **6.12.5.1.2 CIM Requirements**

```
1334 uint16 EnabledState;
1335 uint16 RequestedState;
1336 uint32 CIM_PCIESwitch.RequestStateChange (
1337     [IN] uint16 RequestedState,
1338     [OUT] REF CIM_ConcreteJob Job,
1339     [IN] datetime TimeoutPeriod );
```

1340 **6.12.5.1.3 Behavior Requirements**1341 **6.12.5.1.3.1 Preconditions**

1342 \$instance represents the targeted instance of CIM_PCIESwitch.

```
1343 $instance=<CIM_PCIESwitch single instance>;
```

1344 **6.12.5.1.3.2 Pseudo Code**

```
1345 &smStartRSC ( $instance.getObjectPath() );
1346 &smEnd;
```

1347 **6.12.6 Stop**1348 **6.12.6.1 General Usage of Stop for a Single Property**

1349 This section describes how to implement the `stop` verb when applied to an instance of CIM_PCIESwitch.
 1350 Implementations may support the use of the `stop` verb with CIM_PCIESwitch.

1351 **6.12.6.1.1 Command Form**

```
1352 stop <CIM_PCIESwitch single instance>
```

1353 **6.12.6.1.2 CIM Requirements**

```
1354 uint16 EnabledState;
1355 uint16 RequestedState;
1356 uint32 CIM_PCIESwitch.RequestStateChange (
1357     [IN] uint16 RequestedState,
1358     [OUT] REF CIM_ConcreteJob Job,
1359     [IN] datetime TimeoutPeriod );
```

1360 **6.12.6.1.3 Behavior Requirements**1361 **6.12.6.1.3.1 Preconditions**

1362 \$instance represents the targeted instance of CIM_PCIeSwitch.

```
1363 $instance=<CIM_PCIeSwitch single instance>;
```

1364 **6.12.6.1.3.2 Pseudo Code**

```
1365 &smStopRSC ( $instance.GetObjectPath() );
```

```
1366 &smEnd;
```

1367

**ANNEX A
(informative)**

Change Log

Version	Date	Author	Description
1.0.0	2009-06-04		DMTF Standard Release

1368
1369
1370
1371
1372

1373