Document Number: DSP1022

Date: 2008-10-31

Version: 1.0.0

CPU Profile

Document Type: Specification

Document Status: Final Standard

Document Language: E

CONTENTS

Figures

Tables

Foreword

The *CPU Profile* (DSP1022) was prepared by the Physical Platform Profiles Working Group of the DMTF.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability.

Introduction

The information in this specification should be sufficient for a provider or consumer of this data to identify unambiguously the classes, properties, methods, and values that shall be instantiated and manipulated to represent and manage the processor components of systems and subsystems modeled using the DMTF Common Information Model (CIM) core and extended model definitions.

The target audience for this specification is implementers who are writing CIM-based providers or consumers of management interfaces that represent the component described in this document.

CPU Profile

1 Scope

The *CPU Profile* extends the management capability of referencing profiles by adding the capability to represent CPUs or processors in a managed system. CPU cache memory and associations with CPU physical aspects, as well as profile implementation version information, are modeled in this profile.

2 Normative References

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

2.1 Approved References

DMTF DSP0004, *CIM Infrastructure Specification 2.3.0*

DMTF DSP1033, *Profile Registration Profile 1.0*

DMTF DSP0200, *CIM Operations over HTTP 1.2.0*

DMTF DSP1000, *Management Profile Specification Template*

DMTF DSP1001, *Management Profile Specification Usage Guide*

2.2 References under Development

DMTF DSP0134, *System Management BIOS (SMBIOS) Reference Specification 2.6*

2.3 Other References

ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*

OMG, *Unified Modeling Language (UML) from the Open Management Group (OMG)*

IETF rfc2234, *Augmented BNF for Syntax Specifications: ABNF*

3 Terms and Definitions

For the purposes of this document, the following terms and definitions apply. For the purposes of this document, the terms and definitions in DSP1033 and DSP1001 also apply.

3.1
can
used for statements of possibility and capability, whether material, physical, or causal

3.2
cannot
used for statements of possibility and capability, whether material, physical, or causal

3.3
conditional
indicates requirements to be followed strictly to conform to the document when the specified conditions are met

3.4
mandatory
indicates requirements to be followed strictly to conform to the document and from which no deviation is permitted

3.5
may
indicates a course of action permissible within the limits of the document

3.6
need not
indicates a course of action permissible within the limits of the document

3.7
optional
indicates a course of action permissible within the limits of the document

3.8
referencing profile
indicates a profile that owns the definition of this class and can include a reference to this profile in its “Referenced Profiles” table

3.9
shall
indicates requirements to be followed strictly to conform to the document and from which no deviation is permitted

3.10
shall not
indicates requirements to be followed strictly to conform to the document and from which no deviation is permitted

3.11
should
indicates that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required

3.12
should not
indicates that a certain possibility or course of action is deprecated but not prohibited

3.13
unspecified
indicates that this profile does not define any constraints for the referenced CIM element or operation

3.14
Cache Memory
indicates the instance of CIM_Memory that represents the cache memory for the processor

3.15

Host Processor

indicates the instance of CIM_Processor that represents the processor that hosts the processor core

3.16

Threading Processor Core

indicates the instance of CIM_ProcessorCore that represents the processor core that enables the hardware threading

4 Symbols and Abbreviated Terms

Experimental Maturity Level

Some of the content considered for inclusion in the *CPU Profile* has yet to receive sufficient review to satisfy the adoption requirements set forth by the Technical Committee within the DMTF. This content is presented here as an aid to implementers who are interested in likely future developments within this specification. The content marked experimental may change as implementation experience is gained. There is a high likelihood that it will be included in an upcoming revision of the specification. Until that time, it is purely informational, and is clearly marked within the text.

A sample of the typographical convention for experimental content is included here:

EXPERIMENTAL

Experimental content appears here.

EXPERIMENTAL

5 Synopsis

Profile Name: *CPU*

Version: 1.0.0

Organization: DMTF

CIM Schema Version: 2.19.0

Central Class: CIM_Processor

Scoping Class: CIM_ComputerSystem

The *CPU Profile* is a component profile that extends the management capability of referencing profiles by adding the capability to represent CPUs or processors in a managed system.

Table 1 – Related Profiles

Profile Name	Organization	Version	Requirement	Description
Physical Asset	DMTF	1.0.0	Optional	See section 7.9.
Profile Registration	DMTF	1.0.0	Mandatory	

6 Description

The *CPU Profile* describes CPU or processor devices and associated cache memory used in managed systems.

Figure 1 represents the class schema for the *CPU Profile*. For simplicity, the prefix CIM_ has been removed from the names of the classes.

The CIM_Processor class describes the processor characteristics; the CIM_ProcessorCapabilities class describes the capabilities of the processor. The cores of the processor are represented by the CIM_ProcessorCore class and are associated to the CIM_Processor class through the CIM_ConcreteComponent association. When a core supports hardware threads, the CIM_HardwareThread class is used to represent the thread's properties. The cache memory can be associated with either a processor or a core. The cache memory is represented by the CIM_Memory class and is associated to the CIM_ProcessorCore and CIM_Processor classes through the CIM_AssociatedCacheMemory class. The physical aspects of a processor are represented through an instance of CIM_Chip class. When the cache memory is off-chip/external, the cache memory's physical aspects are represented by an instance of CIM_PhysicalMemory. Also, the class diagram shows the scoping managed system and the profile registration classes.

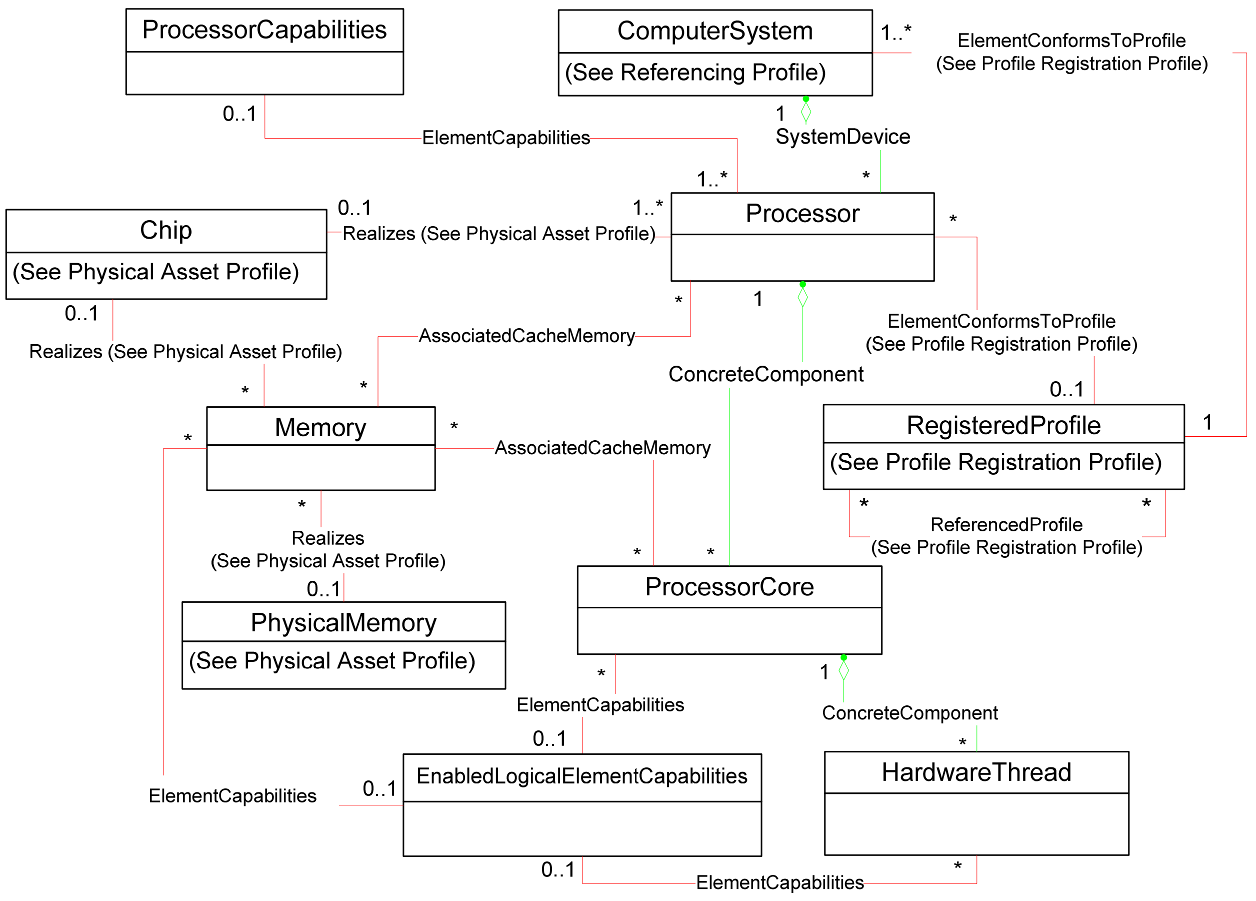


Figure 1 – CPU Profile: Class Diagram

7 Implementation

This section details the requirements related to the arrangement of instances and their properties for implementations of this profile. Methods are listed in section 8 (“Methods”), and properties are listed in section 10 (“CIM Elements”).

7.1 CIM_Processor

Zero or more instances of CIM_Processor shall be instantiated.

7.2 Processor Capabilities

The CIM_ProcessorCapabilities class may be instantiated to represent the processor capabilities. Only one instance of CIM_ProcessorCapabilities shall be associated with a given instance of CIM_Processor through an instance of CIM_ElementCapabilities.

7.2.1 Multi-core or Multi-thread Processor Capabilities

When modeling the capabilities of a multi-core or multi-thread processor, the CIM_ProcessorCapabilities class shall be instantiated and associated to the instance of CIM_Processor that represents the multi-core or multi-thread processor.

The properties of CIM_ProcessorCapabilities described in the “CIM_ProcessorCapabilities Properties” column in Table 2 are defined in terms of the DSP0134 Processor Information structure to provide meaningful context for the interpretation of the properties. The mappings specified in Table 2 shall be used. Note that the underlying represented system does not need to support DSP0134.

Table 2 – CIM_ProcessorCapabilities Properties Mapping to SMBIOS Equivalence

CIM_ProcessorCapabilities Properties	SMBIOS Structure Name	SMBIOS Structure Description
NumberOfProcessorCores	Core Count	Number of cores per processor socket
NumberOfHardwareThreads	Thread Count	Number of threads per processor socket

7.2.2 Single-Core and Single-Thread Processor Capabilities

When modeling the capabilities of a single-core and single-thread processor, the CIM_ProcessorCapabilities may not be instantiated.

When no instance of CIM_ProcessorCapabilities is associated with the instance of CIM_Processor that represents the processor, the processor is a single-core and single-thread processor.

When an instance of CIM_ProcessorCapabilities is associated with the instance of CIM_Processor that represents the single-core and single-thread processor, the following requirements apply:

- The CIM_ProcessorCapabilities.NumberOfProcessorCores property shall have a value of 1.
- The CIM_ProcessorCapabilities.NumberOfHardwareThreads property shall have a value of 1.

7.2.3 CIM_ProcessorCapabilities.RequestedStatesSupported

The RequestedStatesSupported property is an array that contains the supported requested states for the instance of CIM_Processor. This property shall be the super set of the values to be used as the RequestedState parameter in the RequestStateChange( ) method (see section 8.1). The value of the CIM_ProcessorCapabilities.RequestedStatesSupported property shall be an empty array or contain any combination of the following values: 2 (Enabled), 3 (Disabled), or 11 (Reset).

7.2.4 CIM_ProcessorCapabilities.ElementNameEditSupported

The ElementNameEditSupported property shall have a value of TRUE when the implementation supports client modification of the CIM_Processor.ElementName property.

7.2.5 CIM_ProcessorCapabilities.MaxElementNameLen

The MaxElementNameLen property shall be implemented when the ElementNameEditSupported property has a value of TRUE.

7.3 Processor State Management

Processor state management requires that the CIM_Processor.RequestStateChange( ) method be supported (see section 8.1) and the value of the CIM_Processor.RequestedState property not match 12 (Not Applicable).

7.3.1 Processor State Management Support

When the instance of CIM_ProcessorCapabilities does not exist, processor state management shall not be supported.

When the value of the CIM_ProcessorCapabilities.RequestedStatesSupported property of the associated CIM_ProcessorCapabilities instance is an empty array, processor state management shall not be supported.

When the value of the CIM_ProcessorCapabilities.RequestedStatesSupported property of the associated CIM_ProcessorCapabilities instance is not an empty array, processor state management shall be supported.

7.4 CIM_Processor.RequestedState

The CIM_Processor.RequestedState property shall have a value of 12 (Not Applicable) or 5 (No Change), or a value contained in the CIM_ProcessorCapabilities.RequestedStatesSupported property array of the associated CIM_ProcessorCapabilities instance (see section 7.2.2).

When processor state management is supported and the RequestStateChange( ) method is successfully executed, the RequestedState property shall be set to the value of the RequestedState parameter of the RequestStateChange( ) method. After the RequestStateChange( ) method has successfully executed, the RequestedState and EnabledState properties shall have equal values with the exception of the transitional requested state 11 (Reset). The value of the RequestedState property may also change as a result of a request for a change to the processor's enabled state by a non-CIM implementation.

7.4.1 RequestedState—12 (Not Applicable) Value

When processor state management is not supported, the value of the CIM_Processor.RequestedState property shall be 12 (Not Applicable).

7.4.2 RequestedState—5 (No Change) Value

When processor state management is supported, the initial value of the CIM_Processor.RequestedState property shall be 5 (No Change).

7.5 Modeling the Current Enabled State of the Processor

The current enabled state of the processor is described by the CIM_Processor.CPUStatus and CIM_Processor.EnabledState properties. Sections 7.5.1 and 7.5.2 detail the requirements for implementing these two properties.

7.5.1 CIM_Processor.CPUStatus

Table 3 describes the mapping between the values of the CIM_Processor.CPUStatus property and the corresponding description of the state of the processor. The CPUStatus property shall match the values that are specified in Table 3. When the RequestStateChange( ) method executes but does not complete successfully, or the processor is in an indeterminate state, the CPUStatus property shall have value of 0 (Unknown). The value of this property may also change as a result of a change to the processor's enabled state by a non-CIM implementation.

Table 3 – CIM_Processor.CPUStatus Value Descriptions

Value	Description	Extended Description
0	Unknown	Processor state is indeterminate, or the processor state management is not supported.
1	CPU Enabled	Processor shall be enabled.
2	CPU Disabled by User	Processor shall be disabled through client configuration, which may occur through client invocation of the RequestStateChange( ) method or through a non-CIM implementation such as BIOS.
3	CPU Disabled By BIOS (POST Error)	Processor shall be disabled due to a POST error.
4	CPU Is Idle	Processor shall be enabled but idling.

7.5.2 CIM_Processor.EnabledState

The CIM_Processor.EnabledState property shall be implemented in addition to the CIM_Processor.CPUStatus property. When the CPUStatus property has a value that matches a value in the "CPUStatus Value" column in Table 4, the EnabledState property shall have a value that matches a value in the "EnabledState Value" column in the same row in the table.

Table 4 – Mapping for CPUStatus Property and EnabledState Property Values

CPUStatus Value	Description	EnabledState Value	Description
0	Unknown	0 or 5	Unknown or Not Applicable
1	CPU Enabled	2	Enabled
2	CPU Disabled by User	3	Disabled
3	CPU Disabled By BIOS (POST Error)	3	Disabled
4	CPU Is Idle	2	Enabled

7.6 Modeling Individual Processor Cores

Modeling the individual processor cores is optional functionality. When individual processor cores are modeled, the implementation shall instantiate an instance of CIM_ProcessorCore to represent each processor core. All the requirements in this section and its subsections are applicable when an implementation instantiates the CIM_ProcessorCore class.

Each instance of CIM_ProcessorCore shall be associated through an instance of CIM_ConcreteComponent to only one instance of CIM_Processor that represents the processor (Host Processor) that hosts the processor core.

The number of instances of CIM_ProcessorCore associated with the Host Processor shall be equal to the value of the CIM_ProcessorCapabilities.NumberOfProcessorCores property of the instance of CIM_ProcessorCapabilities that is associated with the Host Processor.

7.6.1 Processor Core Capabilities

The CIM_EnabledLogicalElementCapabilities class may be used to model the capabilities of processor cores. When the CIM_EnabledLogicalElementCapabilities class is instantiated to represent the processor core capabilities, the instance of CIM_EnabledLogicalElementCapabilities shall be associated with the CIM_ProcessorCore instance through an instance of CIM_ElementCapabilities and used for advertising the capabilities of the CIM_ProcessorCore instance.

There shall be at most one instance of CIM_EnabledLogicalElementCapabilities associated with a given instance of CIM_ProcessorCore.

7.6.1.1 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported

The RequestedStatesSupported property is an array that contains the supported requested states for the instance of CIM_ProcessorCore. This property shall be the super set of the values to be used as the RequestedState parameter in the RequestStateChange( ) method (see section 8.2). The value of the RequestedStatesSupported property shall be an empty array or contain any combination of the following values: 2 (Enabled), 3 (Disabled), or 11 (Reset).

7.6.1.2 CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported

The ElementNameEditSupported property shall have a value of TRUE when the implementation supports client modification of the CIM_ProcessorCore.ElementName property.

7.6.1.3 CIM_EnabledLogicalElementCapabilities.MaxElementNameLen

The MaxElementNameLen property shall be implemented when the ElementNameEditSupported property has a value of TRUE.

7.6.2 Processor Core State Management

Processor core state management requires that the CIM_ProcessorCore.RequestStateChange( ) method be supported (see section 8.2) and the value of the CIM_ProcessorCore.RequestedState property not match 12 (Not Applicable).

7.6.2.1 Processor Core State Management Support

When no CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_ProcessorCore instance, processor core state management shall not be supported.

When a CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_ProcessorCore instance but the value of the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is an empty array, processor core state management shall not be supported.

When a CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_ProcessorCore instance and the value of the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is not an empty array, processor core state management shall be supported.

7.6.3 CIM_ProcessorCore.RequestedState

The CIM_ProcessorCore.RequestedState property shall have a value of 12 (Not Applicable) or 5 (No Change), or a value contained in the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property array of the associated CIM_EnabledLogicalElementCapabilities instance (see section 7.6.1.1).

When processor core state management is supported and the RequestStateChange( ) method is successfully executed, the RequestedState property shall be set to the value of the RequestedState parameter of the RequestStateChange( ) method. After the RequestStateChange( ) method has successfully executed, the RequestedState and EnabledState properties shall have equal values with the exception of the transitional requested state 11 (Reset). The value of the RequestedState property may also change as a result of a request for a change to the processor's enabled state by a non-CIM implementation.

7.6.3.1 RequestedState—12 (Not Applicable) Value

When processor core state management is not supported, the value of the CIM_ProcessorCore.RequestedState property shall be 12 (Not Applicable).

7.6.3.2 RequestedState—5 (No Change) Value

When processor core state management is supported, the initial value of the CIM_ProcessorCore.RequestedState property shall be 5 (No Change).

7.6.4 Modeling the Current Enabled State of the Processor Core

The current enabled state of the processor core is described by the CIM_ProcessorCore.CoreEnabledState and CIM_ProcessorCore.EnabledState properties. Sections 7.6.4.1 and 7.6.4.2 detail the requirements for implementing these two properties.

7.6.4.1 CIM_ProcessorCore.CoreEnabledState

Table 5 describes the mapping between the values of the CIM_ProcessorCore.CoreEnabledState property and the corresponding description of the state of the processor core. The CoreEnabledState property shall match the values that are specified in Table 5. When the RequestStateChange( ) method executes but does not complete successfully, and the processor core is in an indeterminate state, the CoreEnabledState property shall have a value of 0 (Unknown). The value of this property may also change as a result of a change to the processor's enabled state by a non-CIM implementation.

Table 5 – CIM_ProcessorCore.CoreEnabledState Value Descriptions

Value	Description	Extended Description
0	Unknown	Processor core state is indeterminate, or the processor core state management is not supported.
2	Enabled	Processor core shall be enabled.
3	Disabled	Processor core shall be disabled.
4	Core Disabled User	Processor core shall be disabled through client configuration, which may occur through client invocation of RequestStateChange( ) or through a non-CIM implementation such as BIOS.
5	Core Disabled By Post Error	Processor core shall be disabled due to a POST error.

7.6.4.2 CIM_ProcessorCore.EnabledState

The CIM_ProcessorCore.EnabledState property shall be implemented in addition to the CIM_ProcessorCore.CoreEnabledState property. When the CoreEnabledState property value matches a value in the "CoreEnabledState Value" column in Table 6, the EnabledState property shall have the value that matches the value in the "EnabledState Value" column in the same row in the table.

Table 6 – Mapping for the CoreEnabledState Property and EnabledState Property Values

CoreEnabledState Value	Description	EnabledState Value	Description
0	Unknown	0 or 5	Unknown or Not Applicable
2	Enabled	2	Enabled
3	Disabled	3	Disabled
4	Core Disabled User	3	Disabled
5	Core Disabled By Post Error	3	Disabled

7.7 Modeling Individual Hardware Threads

Modeling the individual hardware threads is optional functionality. When hardware threads are modeled, the implementation shall model processor cores as described in section 7.6 and shall instantiate an instance of CIM_HardwareThread to represent each hardware thread. All the requirements in this section and its subsections are applicable when an implementation instantiates the CIM_HardwareThread class.

The instance of CIM_HardwareThread shall be associated through an instance of CIM_ConcreteComponent to only one instance of CIM_ProcessorCore that represents the processor core that enables the hardware thread (Threading Processor Core).

For a given Host Processor, the number of instances of CIM_HardwareThread that are associated with Threading Processor Cores, which in turn are associated with the Host Processor, shall be equal to the value of the NumberOfHardwareThreads property of the instance of CIM_ProcessorCapabilities that is associated with the Host Processor.

7.7.1 Hardware Thread Capabilities

When the CIM_EnabledLogicalElementCapabilities class is instantiated to represent the hardware thread capabilities, the instance of CIM_EnabledLogicalElementCapabilities shall be associated with the CIM_HardwareThread instance through an instance of CIM_ElementCapabilities and used for advertising the capabilities of the CIM_HardwareThread instance.

At most one instance of CIM_EnabledLogicalElementCapabilities shall be associated with a given instance of CIM_HardwareThread.

7.7.1.1 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported

The RequestedStatesSupported property is an array that contains the supported requested states for the instance of CIM_HardwareThread. This property shall be the super set of the values to be used as the RequestedState parameter in the RequestStateChange( ) method (see section 8.3). The value of the RequestedStatesSupported property shall be an empty array or contain any combination of the following values: 2 (Enabled), 3 (Disabled), or 11 (Reset).

7.7.1.2 CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported

The ElementNameEditSupported property shall have a value of TRUE when the implementation supports client modification of the CIM_HardwareThread.ElementName property.

7.7.1.3 CIM_EnabledLogicalElementCapabilities.MaxElementNameLen

The MaxElementNameLen property shall be implemented when the ElementNameEditSupported property has a value of TRUE.

7.7.2 Hardware Thread State Management

Hardware thread state management requires that the CIM_HardwareThread.RequestStateChange( ) method be supported (see section 8.3) and the value of the CIM_HardwareThread.RequestedState property not match 12 (Not Applicable).

7.7.2.1 Hardware Thread State Management Support

When no CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_HardwareThread instance, hardware thread state management shall not be supported.

When a CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_HardwareThread instance but the value of the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is an empty array, hardware thread state management shall not be supported.

When a CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_HardwareThread instance and the value of the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is not an empty array, hardware thread state management shall be supported.

7.7.3 CIM_HardwareThread.RequestedState

The CIM_HardwareThread.RequestedState property shall have a value of 12 (Not Applicable) or 5 (No Change), or a value contained in the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property array of the associated CIM_EnabledLogicalElementCapabilities instance (see section 7.7.1.1).

When hardware thread state management is supported and the RequestStateChange( ) method is successfully executed, the RequestedState property shall be set to the value of the RequestedState parameter of the RequestStateChange( ) method. After the RequestStateChange( ) method has successfully executed, the RequestedState and EnabledState properties shall have equal values with the exception of the transitional requested state 11 (Reset). The value of the RequestedState property may also change as a result of a request for a change to the hardware thread’s enabled state by a non-CIM implementation.

7.7.3.1 RequestedState—12 (Not Applicable) Value

When hardware thread state management is not supported, the value of the CIM_HardwareThread.RequestedState property shall be 12 (Not Applicable).

7.7.3.2 RequestedState—5 (No Change) Value

When hardware thread state management is supported, the initial value of the CIM_HardwareThread.RequestedState property shall be 5 (No Change).

7.7.4 CIM_HardwareThread.EnabledState

Table 7 describes the mapping between the values of the CIM_HardwareThread.EnabledState property and the corresponding description of the state of the hardware thread. The EnabledState property shall match the values that are specified in Table 7. When the RequestStateChange( ) method executes but does not complete successfully, and the hardware thread is in an indeterminate state, the EnabledState property shall have a value of 5 (Not Applicable). The value of this property may also change as a result of a change to the hardware thread's enabled state by a non-CIM implementation.

Table 7 – CIM_HardwareThread.EnabledState Value Descriptions

Value	Description	Extended Description
2	Enabled	Hardware thread shall be enabled.
3	Disabled	Hardware thread shall be disabled.
5	Not Applicable	Hardware thread state is indeterminate, or hardware thread state management is not supported.

7.8 Modeling Cache Memory

Modeling the cache memory of the processor is optional. The implementation may instantiate instances of CIM_Memory to represent the cache memory. All the requirements in this section and its subsections are applicable when an implementation instantiates the CIM_Memory class that represents cache memory.

A single instance of CIM_Memory shall exist for each discrete cache memory. When the cache memory is shared, the single instance of CIM_Memory shall be associated with multiple instances of CIM_Processor or CIM_ProcessorCore. When the cache memory is not shared, the instance of CIM_Memory shall be associated with exactly one instance of CIM_Processor or CIM_ProcessorCore.

When the optional behavior described in section 7.6 is implemented, each instance of CIM_Memory that represents the cache memory used by the processor core shall be associated with the instance of CIM_ProcessorCore that represents the processor core through an instance of CIM_AssociatedCacheMemory and shall not be associated with the Host Processor of the core.

When the optional behavior described in section 7.6 is not implemented, each instance of CIM_Memory that represents the cache memory used by the processor shall be associated through an instance of the CIM_AssociatedCacheMemory to the instance of CIM_Processor.

7.8.1 Cache Memory Capabilities

When the CIM_EnabledLogicalElementCapabilities class is instantiated to represent the cache memory capabilities, the instance of CIM_EnabledLogicalElementCapabilities shall be associated with the CIM_Memory instance through an instance of CIM_ElementCapabilities and used for advertising the capabilities of the CIM_Memory instance.

At most one instance of CIM_EnabledLogicalElementCapabilities shall be associated with a given instance of CIM_Memory.

7.8.1.1 CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported

The RequestedStatesSupported property is an array that contains the supported requested states for the instance of CIM_Memory. This property shall be the super set of the values to be used as the RequestedState parameter in the RequestStateChange( ) method (see section 8.4). The value of the RequestedStatesSupported property shall be an empty array or contain any combination of the following values: 2 (Enabled), 3 (Disabled), or 11 (Reset).

7.8.1.2 CIM_EnabledLogicalElementCapabilities.ElementNameEditSupported

The ElementNameEditSupported property shall have a value of TRUE when the implementation supports client modification of the CIM_Memory.ElementName property.

7.8.1.3 CIM_EnabledLogicalElementCapabilities.MaxElementNameLen

The MaxElementNameLen property shall be implemented when the ElementNameEditSupported property has a value of TRUE.

7.8.2 Cache Memory State Management

Cache memory state management requires that the CIM_Memory.RequestStateChange( ) method be supported (see section 8.4) and the value of the CIM_Memory.RequestedState property not match 12 (Not Applicable).

7.8.2.1 Cache Memory State Management Support

When no CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_Memory instance, cache memory state management shall not be supported.

When a CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_Memory instance but the value of the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is an empty array, cache memory state management shall not be supported.

When a CIM_EnabledLogicalElementCapabilities instance is associated with the CIM_Memory instance and the value of the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property is not an empty array, cache memory state management shall be supported.

7.8.3 CIM_Memory.RequestedState

The CIM_Memory.RequestedState property shall have a value of 12 (Not Applicable) or 5 (No Change), or a value contained in the CIM_EnabledLogicalElementCapabilities.RequestedStatesSupported property array of the associated CIM_EnabledLogicalElementCapabilities instance (see section 7.8.1.1).

When cache memory state management is supported and the RequestStateChange( ) method is successfully executed, the RequestedState property shall be set to the value of the RequestedState parameter of the RequestStateChange( ) method. After the RequestStateChange( ) method has successfully executed, the RequestedState and EnabledState properties shall have equal values with the exception of the transitional requested state 11 (Reset). The value of the RequestedState property may also change as a result of a request for a change to the cache memory’s enabled state by a non-CIM implementation.

7.8.3.1 RequestedState—12 (Not Applicable) Value

When cache memory state management is not supported, the value of the CIM_Memory.RequestedState property shall be 12 (Not Applicable).

7.8.3.2 RequestedState—5 (No Change) Value

When cache memory state management is supported, the initial value of the CIM_Memory.RequestedState property shall be 5 (No Change).

7.8.4 CIM_Memory.EnabledState

Table 8 describes the mapping between the values of the CIM_Memory.EnabledState property and the corresponding description of the state of the cache memory. The EnabledState property shall match the values that are specified in Table 8. When the RequestStateChange( ) method executes but does not complete successfully, and the cache memory is in an indeterminate state, the EnabledState property shall have value of 5 (Not Applicable). The value of this property may also change as a result of a change to the cache memory's enabled state by a non-CIM implementation.

Table 8 – CIM_Memory.EnabledState Value Descriptions

Value	Description	Extended Description
2	Enabled	Cache memory shall be enabled.
3	Disabled	Cache memory shall be disabled.
5	Not Applicable	Cache memory state is indeterminate, or cache memory state management is not supported.

7.9 Modeling Physical Aspects of Processor and Cache Memory

The *Physical Asset Profile* may be implemented to model the physical aspects of a processor, including the asset information of the processor or the internal or off-chip cache memory.

When the processor's or internal cache memory's physical aspects are represented, a CIM_Chip instance shall be instantiated and associated with the instance of CIM_Processor or with any instances of CIM_Memory that represent the internal cache through instances of CIM_Realizes.

When the off-chip cache memory is represented along with its physical aspects, a CIM_PhysicalMemory instance shall be instantiated and associated with the instance of CIM_Memory through an instance of CIM_Realizes.

Note that when processor cores or hardware threads for the processor are modeled with the physical aspects of the processor, the instances of CIM_ProcessorCore and CIM_HardwareThread shall not be associated with the instance of CIM_Chip that represents the physical aspects of the processor.

The configuration capacity of the managed system for processors may be modeled using the optional behavior specified in the "Modeling Configuration Capacity" section of the *Physical Asset Profile*.

8 Methods

This section details the requirements for supporting intrinsic operations and extrinsic methods for the CIM elements defined by this profile.

8.1 CIM_Processor.RequestStateChange( )

Invocation of the CIM_Processor.RequestStateChange( ) method changes the element's state to the value that is specified in the RequestedState parameter.

Return code values for the RequestStateChange( ) method shall be as specified in Table 9. Parameters of the RequestStateChange( ) method are specified in Table 10.

When processor state management is supported, the RequestStateChange( ) method shall be implemented and shall not return a value of 1 (Not Supported) (see section 7.3.1).

Invoking the CIM_Processor.RequestStateChange( ) method multiple times could result in earlier requests being overwritten or lost.

No standard messages are defined for this method.

Table 9 – CIM_Processor.RequestStateChange( ) Method: Return Code Values

Value	Description
0	Request was successfully executed.
1	Method is not supported in the implementation.
2	Error occurred
4096	Job started

Table 10 – CIM_Processor.RequestStateChange( ) Method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	RequestedState	uint16	Valid state values: 2 (Enabled) 3 (Disabled) 11 (Reset)
OUT	Job	CIM_ConcreteJob REF	Returned if job started
IN, REQ	TimeoutPeriod	datetime	Client-specified maximum amount of time the transition to a new state is supposed to take: 0 or NULL – No time requirements <interval> – Maximum time allowed

8.2 CIM_ProcessorCore.RequestStateChange( )

Invocation of the CIM_ProcessorCore.RequestStateChange( ) method changes the element’s state to the value that is specified in the RequestedState parameter.

Return code values for the RequestStateChange( ) method shall be as specified in Table 11. Parameters of the RequestStateChange( ) method are specified in Table 12.

When processor core state management is supported, the RequestStateChange( ) method shall be implemented and shall not return a value of 1 (Not Supported) (see section 7.6.2.1).

Invoking the CIM_ProcessorCore.RequestStateChange( ) method multiple times could result in earlier requests being overwritten or lost.

No standard messages are defined for this method.

Table 11 – CIM_ProcessorCore.RequestStateChange( ) Method: Return Code Values

Value	Description
0	Request was successfully executed.
1	Method is not supported in the implementation.
2	Error occurred
4096	Job started

Table 12 – CIM_ProcessorCore.RequestStateChange( ) Method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	RequestedState	uint16	Valid state values: 2 (Enabled) 3 (Disabled) 11 (Reset)
OUT	Job	CIM_ConcreteJob REF	Returned if job started
IN, REQ	TimeoutPeriod	datetime	Client-specified maximum amount of time the transition to a new state is supposed to take: 0 or NULL – No time requirements <interval> – Maximum time allowed

8.3 CIM_HardwareThread.RequestStateChange( )

Invocation of the CIM_HardwareThread.RequestStateChange( ) method changes the element's state to the value that is specified in the RequestedState parameter.

Return code values for the RequestStateChange( ) method shall be as specified in Table 13. Parameters of the RequestStateChange( ) method are specified in Table 14.

When hardware thread state management is supported, the RequestStateChange( ) method shall be implemented and shall not return a value of 1 (Not Supported) (see section 7.7.2.1).

Invoking the CIM_HardwareThread.RequestStateChange( ) method multiple times could result in earlier requests being overwritten or lost.

No standard messages are defined for this method.

Table 13 – CIM_HardwareThread.RequestStateChange( ) Method: Return Code Values

Value	Description
0	Request was successfully executed.
1	Method is not supported in the implementation.
2	Error occurred
4096	Job started

Table 14 – CIM_HardwareThread.RequestStateChange( ) Method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	RequestedState	uint16	Valid state values: 2 (Enabled) 3 (Disabled) 11 (Reset)
OUT	Job	CIM_ConcreteJob REF	Returned if job started
IN, REQ	TimeoutPeriod	datetime	Client-specified maximum amount of time the transition to a new state is supposed to take: 0 or NULL – No time requirements <interval> – Maximum time allowed

8.4 CIM_Memory.RequestStateChange( )

Invocation of the CIM_Memory.RequestStateChange( ) method changes the element's state to the value that is specified in the RequestedState parameter.

Return code values for the RequestStateChange( ) method shall be as specified in Table 15. Parameters of the RequestStateChange( ) method are specified in Table 16.

When memory state management is supported, the RequestStateChange( ) method shall be implemented and shall not return a value of 1 (Not Supported) (see section 7.8.2.1).

Invoking the CIM_Memory.RequestStateChange( ) method multiple times could result in earlier requests being overwritten or lost.

No standard messages are defined for this method.

Table 15 – CIM_Memory.RequestStateChange( ) Method: Return Code Values

Value	Description
0	Request was successfully executed.
1	Method is not supported in the implementation.
2	Error occurred
4096	Job started

Table 16 – CIM_Memory.RequestStateChange( ) Method: Parameters

Qualifiers	Name	Type	Description/Values
IN, REQ	RequestedState	uint16	Valid state values: 2 (Enabled) 3 (Disabled) 11 (Reset)
OUT	Job	CIM_ConcreteJob REF	Returned if job started
IN, REQ	TimeoutPeriod	datetime	Client-specified maximum amount of time the transition to a new state is supposed to take: 0 or NULL – No time requirements <interval> – Maximum time allowed

8.5 Profile Conventions for Operations

Support for operations for each profile class (including associations) is specified in the following subclauses. Each subclause includes either the statement "All operations in the default list in section 8.5 are supported as described by DSP0200 version 1.2" or a table listing all of the operations that are not supported by this profile or where the profile requires behavior other than that described by DSP0200 version 1.2.

The default list of operations is as follows:

- GetInstance
- EnumerateInstances
- EnumerateInstanceNames
- Associators

- AssociatorNames
- References
- ReferenceNames

A compliant implementation shall support all the operations in the default list for each class, unless the "Requirement" column states something other than *Mandatory*.

8.6 CIM_AssociatedCacheMemory

Table 17 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 17 – Operations: CIM_AssociatedCacheMemory

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.7 CIM_ConcreteComponent—References CIM_HardwareThread and CIM_Processor

Table 18 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 18 – Operations: CIM_ConcreteComponent

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.8 CIM_ConcreteComponent—References CIM_ProcessorCore and CIM_Processor

Table 19 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 19 – Operations: CIM_ConcreteComponent

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.9 CIM_ElementCapabilities—References CIM_HardwareThread and CIM_EnabledLogicalElementCapabilities

Table 20 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 20 – Operations: CIM_ElementCapabilities

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.10 CIM_ElementCapabilities—References CIM_Memory and CIM_EnabledLogicalElementCapabilities

Table 21 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 21 – Operations: CIM_ElementCapabilities

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.11 CIM_ElementCapabilities—References CIM_Processor and CIM_ProcessorCapabilities

Table 22 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 22 – Operations: CIM_ElementCapabilities

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.12 CIM_ElementCapabilities—References CIM_ProcessorCore and CIM_EnabledLogicalElementCapabilities

Table 23 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 23 – Operations: CIM_ElementCapabilities

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

8.13 CIM_EnabledLogicalElementCapabilities

All operations in the default list in section 8.5 are supported as described by DSP0200 version 1.2.

8.14 CIM_HardwareThread

Table 24 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 24 – Operations: CIM_HardwareThread

Operation	Requirement	Messages
ModifyInstance	Optional. See section 8.14.1.	None

8.14.1 CIM_HardwareThread—ModifyInstance

This section details the requirements for the ModifyInstance operation applied to an instance of CIM_HardwareThread. The ModifyInstance operation may be supported.

The ModifyInstance operation shall be supported and the CIM_HardwareThread.ElementName property shall be modifiable when the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance that is associated with the CIM_HardwareThread instance has a value of TRUE. See section 7.7.1.2.

8.15 CIM_Memory

Table 25 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 25 – Operations: CIM_Memory

Operation	Requirement	Messages
ModifyInstance	Optional. See section 8.15.1.	None

8.15.1 CIM_Memory—ModifyInstance

This section details the requirements for the ModifyInstance operation applied to an instance of CIM_Memory. The ModifyInstance operation may be supported.

The ModifyInstance operation shall be supported and the CIM_Memory.ElementName property shall be modifiable when the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance that is associated with the CIM_Memory instance has a value of TRUE. See section 7.8.1.2.

8.16 CIM_Processor

Table 26 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 26 – Operations: CIM_Processor

Operation	Requirement	Messages
ModifyInstance	Optional. See section 8.16.1.	None

8.16.1 CIM_Processor—ModifyInstance

This section details the requirements for the ModifyInstance operation applied to an instance of CIM_Processor. The ModifyInstance operation may be supported.

The ModifyInstance operation shall be supported and the CIM_Processor.ElementName property shall be modifiable when the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance that is associated with the CIM_Processor instance has a value of TRUE. See section 7.2.4.

8.17 CIM_ProcessorCapabilities

All operations in the default list in section 8.5 are supported as described by DSP0200 version 1.2.

8.18 CIM_ProcessorCore

Table 27 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 27 – Operations: CIM_ProcessorCore

Operation	Requirement	Messages
ModifyInstance	Optional. See section 8.18.1.	None

8.18.1 CIM_ProcessorCore—ModifyInstance

This section details the requirements for the ModifyInstance operation applied to an instance of CIM_ProcessorCore. The ModifyInstance operation may be supported.

The ModifyInstance operation shall be supported and the CIM_ProcessorCore.ElementName property shall be modifiable when the ElementNameEditSupported property of the CIM_EnabledLogicalElementCapabilities instance that is associated with the CIM_ProcessorCore instance has a value of TRUE. See section 7.6.1.2.

8.19 CIM_SystemDevice

Table 28 lists operations that either have special requirements beyond those from DSP0200 version 1.2 or shall not be supported.

Table 28 – Operations: CIM_SystemDevice

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

9 Use Cases

This section contains object diagrams and use cases for the *CPU Profile*.

9.1 Object Diagrams

Figure 2 represents a possible instantiation of the *CPU Profile*. In this instantiation, cpu1 belongs to system1. The capabilities of cpu1 are represented with capabilities1. cpu1 has cache memory represented by memory1. The *CPU Profile* implementation and versioning information is advertised through profile2.

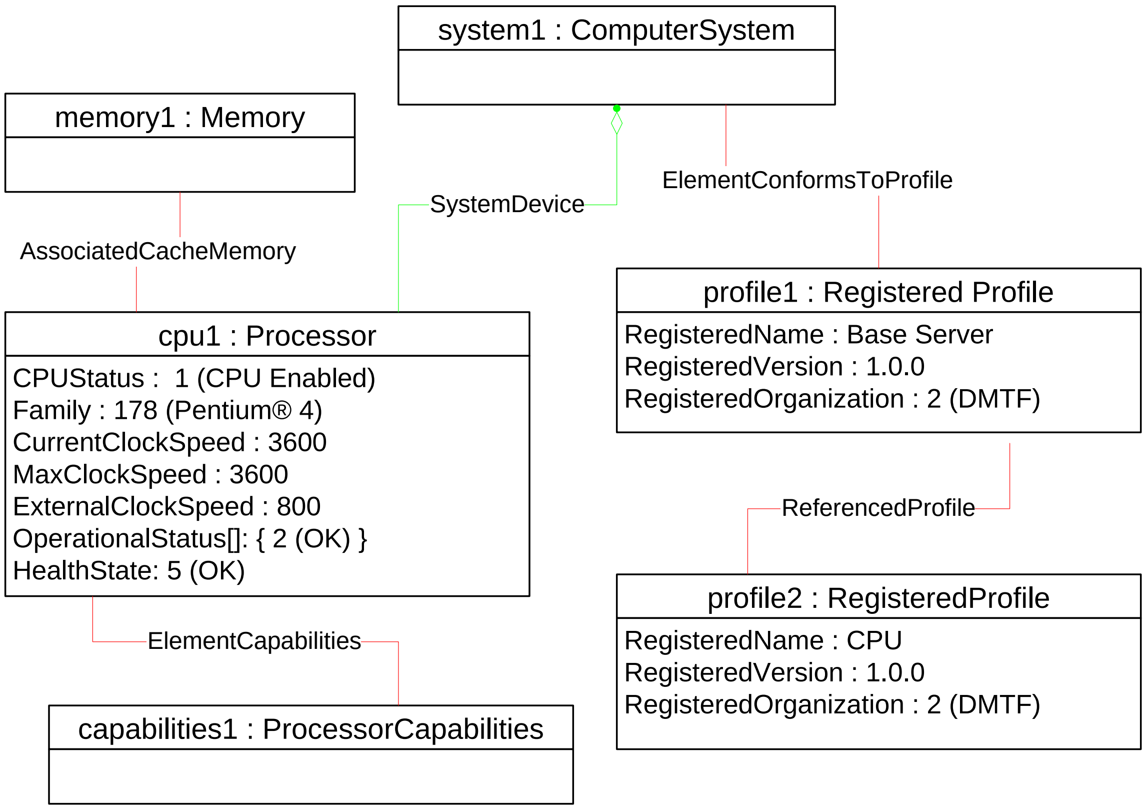

system1 : ComputerSystem
memory1 : Memory
SystemDevice
ElementConformsToProfile
AssociatedCacheMemory
profile1 : Registered Profile
RegisteredName : Base Server
RegisteredVersion : 1.0.0
RegisteredOrganization : 2 (DMTF)
cpu1 : Processor
CPUStatus : 1 (CPU Enabled)
Family : 178 (Pentium® 4)
CurrentClockSpeed : 3600
MaxClockSpeed : 3600
ExternalClockSpeed : 800
OperationalStatus[]: { 2 (OK) }
HealthState: 5 (OK)
ReferencedProfile
profile2 : RegisteredProfile
RegisteredName : CPU
RegisteredVersion : 1.0.0
RegisteredOrganization : 2 (DMTF)
ElementCapabilities
capabilities1 : ProcessorCapabilities

Figure 2 – Registered Profile

Figure 3 represents a possible instantiation of the *CPU Profile* representing a dual core processor, cpu1. The individual cores and hardware threads of cpu1 are not represented, but capabilities1 advertises that cpu1 is a dual core processor capable of two hardware threads, one thread per each core. If system1 supports *SMBIOS Reference Specification 2.6* or later, the value of the NumberOfProcessorCores property will be equal to the SMBIOS Processor Information structure's Core Count structure value, and the value of the NumberOfHardwareThreads property will be equal to the SMBIOS Processor Information structure's Thread Count structure value. memory1 and memory2 are the cache memories of cpu1. Memory1 represents the level 1 cache, and memory2 is the level 2 cache, as denoted by the instances of CIM_AssociatedCacheMemory that associate memory1 and memory2 with cpu1. The physical aspects of cpu1 are represented by chip1, associated to cpu1 through an instance of CIM_Realizes.

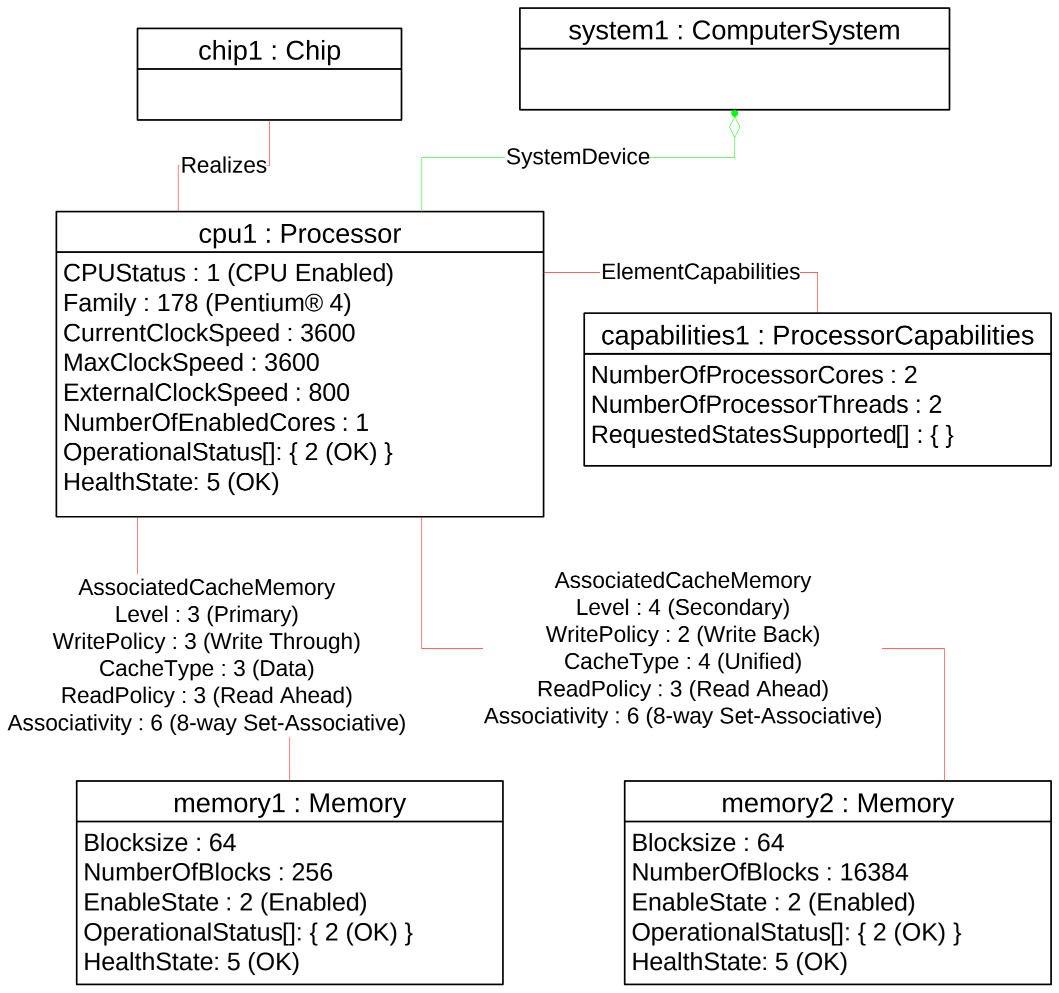

chip1 : Chip
system1 : ComputerSystem
Realizes
SystemDevice
cpu1 : Processor
CPUStatus : 1 (CPU Enabled)
Family : 178 (Pentium® 4)
CurrentClockSpeed : 3600
MaxClockSpeed : 3600
ExternalClockSpeed : 800
NumberOfEnabledCores : 1
OperationalStatus[]: { 2 (OK) }
HealthState: 5 (OK)
ElementCapabilities
capabilities1 : ProcessorCapabilities
NumberOfProcessorCores : 2
NumberOfProcessorThreads : 2
RequestedStatesSupported[] : { }
AssociatedCacheMemory
Level : 3 (Primary)
WritePolicy : 3 (Write Through)
CacheType : 3 (Data)
ReadPolicy : 3 (Read Ahead)
Associativity : 6 (8-way Set-Associative)
AssociatedCacheMemory
Level : 4 (Secondary)
WritePolicy : 2 (Write Back)
CacheType : 4 (Unified)
ReadPolicy : 3 (Read Ahead)
Associativity : 6 (8-way Set-Associative)
memory1 : Memory
Blocksize : 64
NumberOfBlocks : 256
EnableState : 2 (Enabled)
OperationalStatus[]: { 2 (OK) }
HealthState: 5 (OK)
memory2 : Memory
Blocksize : 64
NumberOfBlocks : 16384
EnableState : 2 (Enabled)
OperationalStatus[]: { 2 (OK) }
HealthState: 5 (OK)

Figure 3 – Multi-core CPU

Figure 4 represents a possible instantiation of the *CPU Profile* representing a dual core processor, cpu1. Each of the processor cores is represented by an instance of CIM_ProcessorCore: core1 and core2, associated to the Host Processor, cpu1, through instances of CIM_ConcreteComponent. Each of the cores has one hardware thread, represented by thread1 and thread2, associated with it through instances of CIM_ConcreteComponent. The cache memories, memory1 and memory2, are associated to the processor cores that use them. Based on the capabilities of core1 and core2, represented by capabilities2, both processor cores can be enabled or disabled using the RequestStateChange( ) method. Figure 5 shows the same instantiation of *CPU Profile* after the RequestStateChange( ) method on core2 has successfully executed.

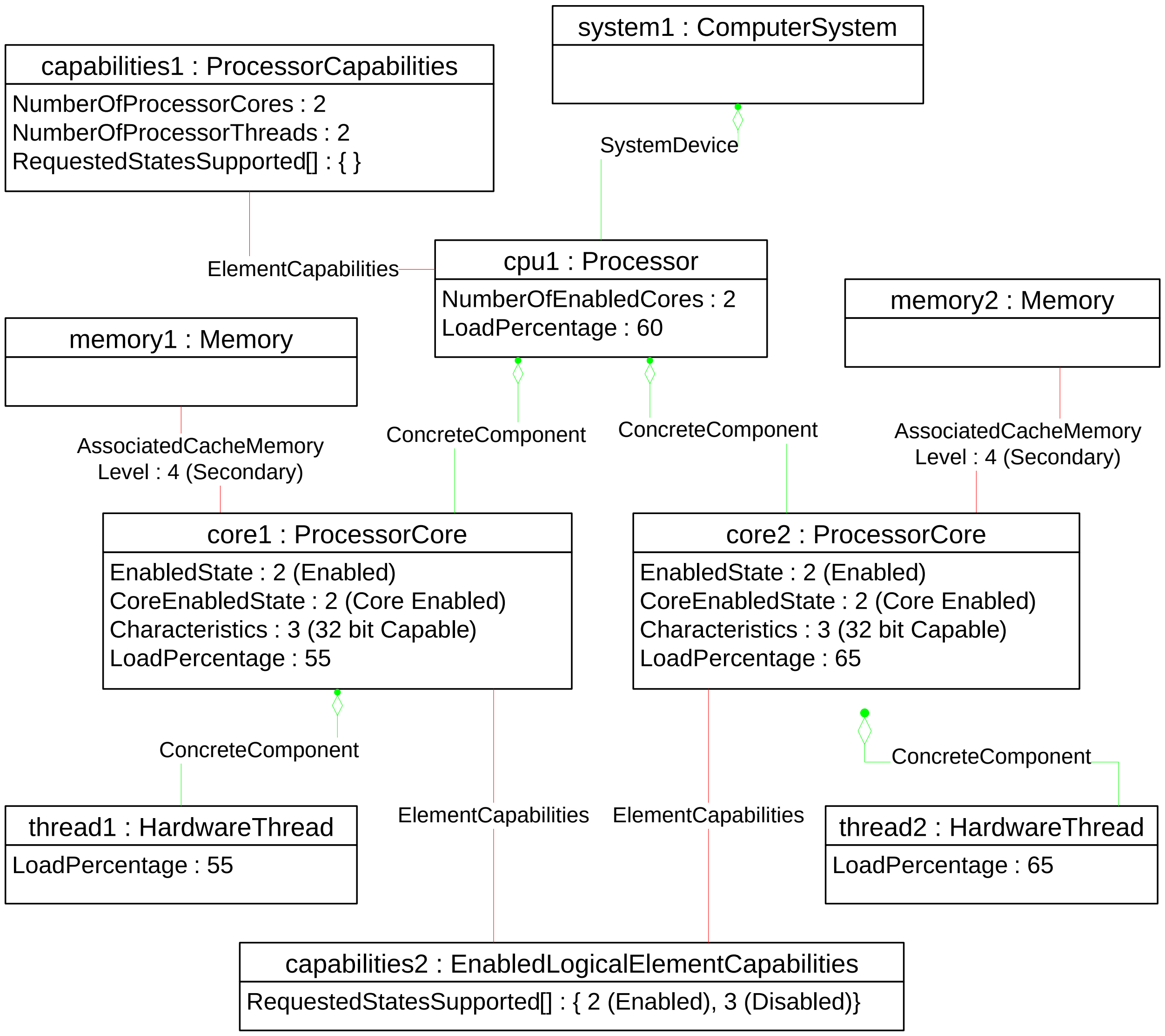


Figure 4 – Detailed Multi-core CPU

Figure 5 represents a possible instantiation of the *CPU Profile* in which one of the cores of a dual core processor, cpu1, has been disabled by the user using the RequestStateChange( ) method. core2's EnabledState property has value of 3 (Disabled) and the CoreEnabledState property has value 4 (Core Disabled by User).

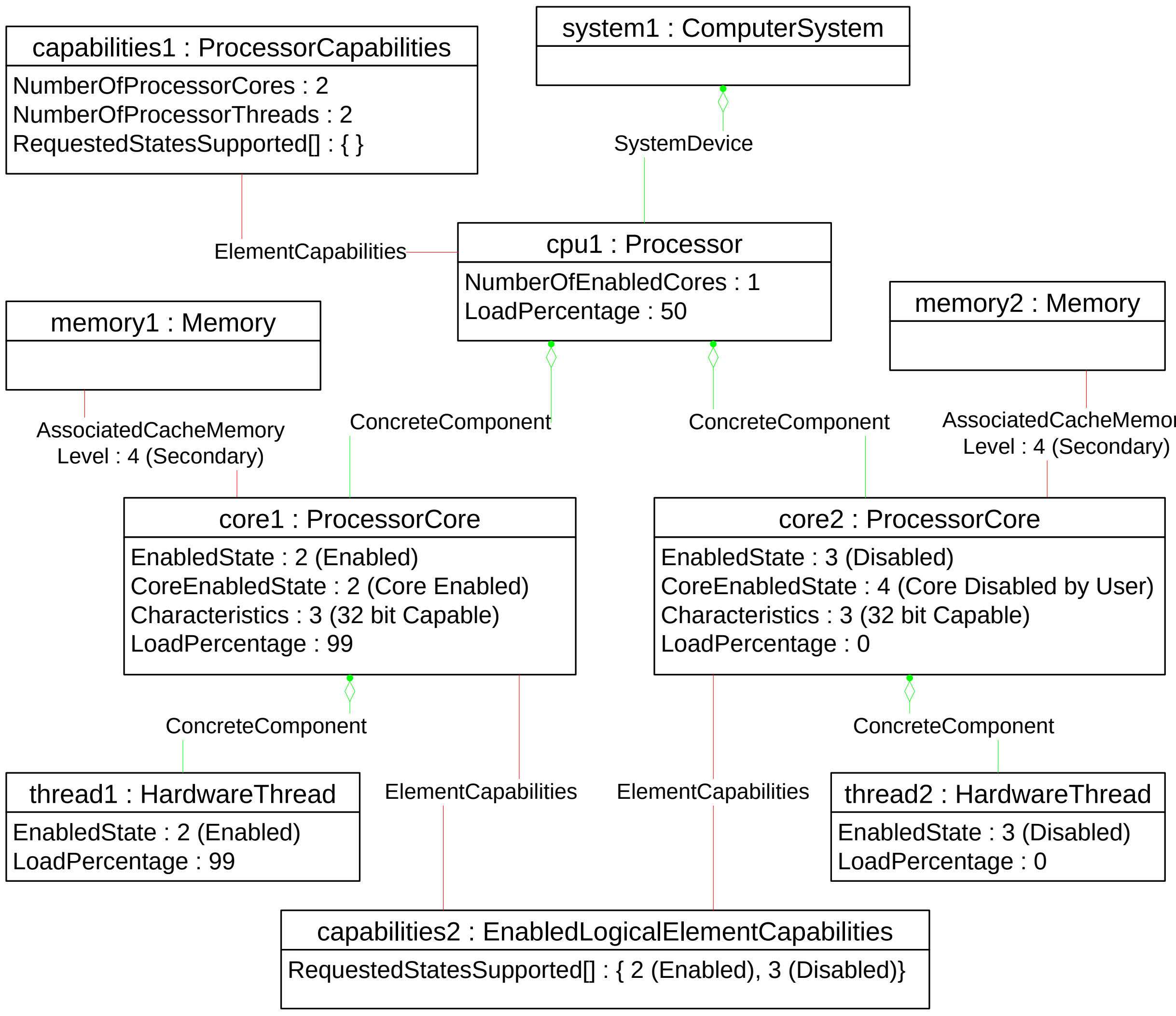


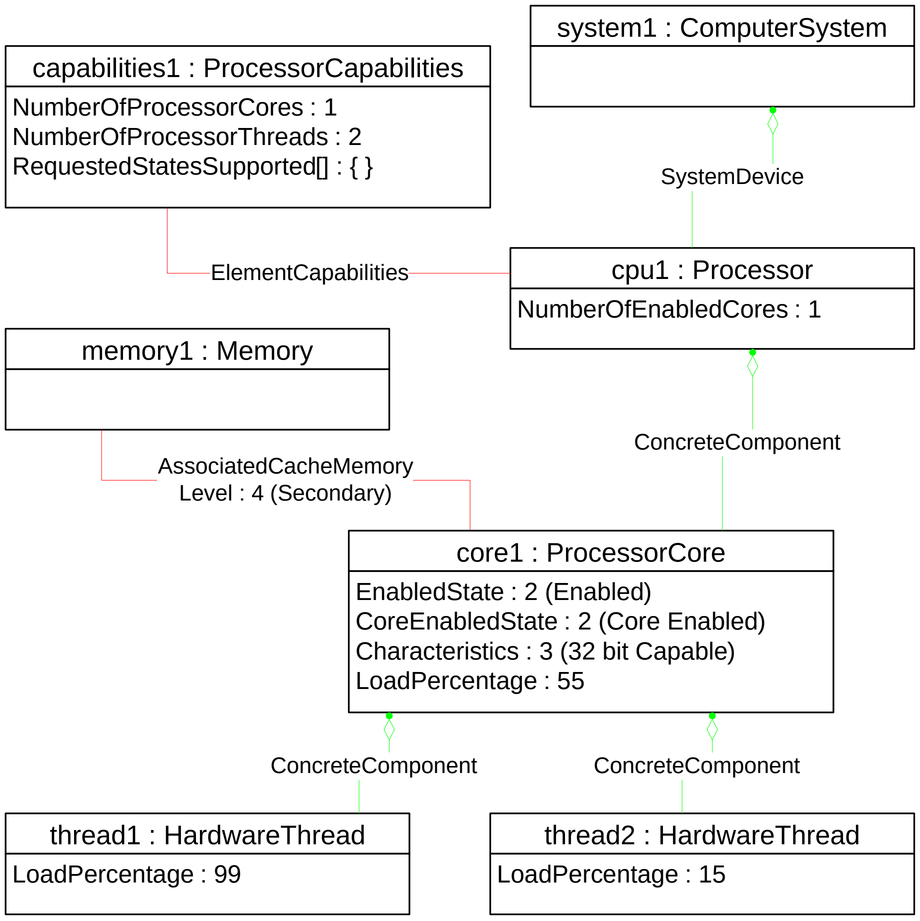
Figure 5 – Multi-core CPU with a Disabled Core

Figure 6 represents a possible instantiation of the *CPU Profile* representing a single core processor with multiple threads. thread1 and thread2 represent the hardware threads that exist on core1 and are associated to core1 through an instance of CIM_ConcreteComponent. cpu1 advertises the capabilities of multiple hardware threads through the capabilities1 NumberOfProcessorThreads property. The cache memory, memory1, is associated to core1, which is using the cache memory.

EXPERIMENTAL

The load percentage calculation is implementation specific; the LoadPercentage property value for core1 is calculated by taking the average of the values of the LoadPercentage property of thread1 and thread2.

EXPERIMENTAL



819

820 **Figure 6 – Single Core, Multi-hardware Thread CPU**

Figure 7 represents another instantiation of the *CPU Profile*. In this case, cpu1's cache memory, memory1, has a separate package represented by pmem1 and associated to memory1 through an instance of CIM_Realizes. The existence of pmem1 associated with the cpu1's cache memory indicates that the processor uses off-chip cache memory.

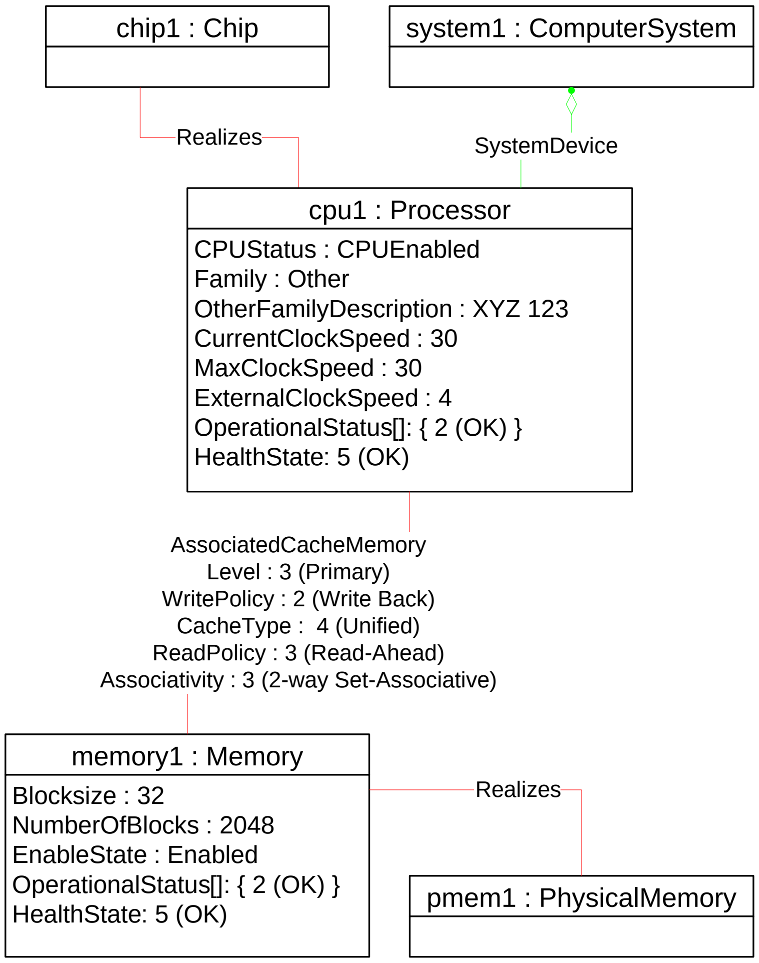


Figure 7 – Processor with Off-Chip Cache

9.2 Change the Enabled State of the Memory to the Desired State

A client can change the enabled state of the memory as follows:

1) Select the instance of CIM_Memory.
2) Select the associated instance of CIM_EnabledLogicalElementCapabilities and verify whether the RequestedStatesSupported property contains the desired state.
3) If the RequestedStatesSupported property contains the desired state, select the instance of CIM_Memory and execute the RequestStateChange( ) method with the desired state as a RequestedState parameter.

After the successful execution of the method, the EnabledState property of the instance of CIM_Memory will have the value of the desired state.

9.3 Change the Enabled State of the CPU to the Desired State

A client can change the enabled state of the CPU as follows:

1. Select the instance of CIM_Processor.
2. Select the associated instance of CIM_ProcessorCapabilities and verify whether the RequestedStatesSupported property contains the desired state.
3. If the RequestedStatesSupported property contains the desired state, select the instance of CIM_Processor and execute the RequestStateChange( ) method with the desired state as a RequestedState parameter.

After the successful execution of the method, the EnabledState property of the instance of CIM_Processor will have the value of the desired state.

9.4 Change the Enabled State of the CPU's Core to the Desired State

A client can change the enabled state of the CPU's core as follows:

1. Select the instance of CIM_ProcessorCore.
2. Select the associated instance of CIM_EnabledLogicalElementCapabilities and verify whether the RequestedStatesSupported property contains the desired state.
3. If the RequestedStatesSupported property contains the desired state, select the instance of CIM_ProcessorCore and execute the RequestStateChange( ) method with the desired state as a RequestedState parameter.

After the successful execution of the method, the EnabledState property of the instance of CIM_ProcessorCore will have the value of the desired state.

9.5 Change the Enabled State of the CPU's Hardware Thread to the Desired State

A client can change the enabled state of the CPU's hardware thread as follows:

1. Select the instance of CIM_HardwareThread.
2. Select the associated instance of CIM_EnabledLogicalElementCapabilities and verify whether the RequestedStatesSupported property contains the desired state.
3. If the RequestedStatesSupported property contains the desired state, select the instance of CIM_ProcessorThread and execute the RequestStateChange( ) method with the desired state as a RequestedState parameter.

After the successful execution of the method, the EnabledState property of the instance of CIM_HardwareThread will have the value of the desired state.

9.6 Retrieve All the Processor Cores for the CPU

A client can retrieve all of the processor cores for the CPU by selecting all the CIM_ProcessorCore instances that are associated with the given instance of CIM_Processor through instances of CIM_Component.

9.7 Retrieve All the Hardware Threads for the CPU

A client can retrieve all of the hardware threads for the CPU as follows:

1. Select all the CIM_ProcessorCore instances that are associated with the given instance of CIM_Processor through instances of CIM_Component.

2. For each instance of CIM_ProcessorCore, select the instances of CIM_HardwareThread that are associated through instances of CIM_Component.

9.8 Retrieve CPU's Cache Memory Information for the CPU

A client can retrieve the CPU's cache memory information as follows:

1. Select all the instances of CIM_ProcessorCore that are associated with the given instance of CIM_Processor through instances of CIM_Component.
2. If no instance of CIM_ProcessorCore exists, select the instances of CIM_AssociatedCacheMemory that reference the given instance of CIM_Processor, as well as all the instances of CIM_Memory that are associated with the given instance of CIM_Processor through instances of CIM_AssociatedCacheMemory.
3. Otherwise, for each instance of CIM_ProcessorCore, select the instances of CIM_AssociatedCacheMemory that reference the instance of CIM_ProcessorCore, as well as all the instances of CIM_Memory that are associated with the instance of CIM_ProcessorCore through instances of CIM_AssociatedCacheMemory.

10 CIM Elements

Table 29 shows the instances of CIM Elements for this profile. Instances of the CIM Elements shall be implemented as described in Table 29. Sections 7 ("Implementation") and 8 ("Methods") may impose additional requirements on these elements.

Table 29 – CIM Elements: CPU Profile

Element Name	Requirement	Description
Classes		
CIM_AssociatedCacheMemory	Optional	See sections 10.1 and 7.8.
CIM_ConcreteComponent (references CIM_HardwareThread and CIM_ProcessorCore)	Optional	See section 10.2.
CIM_ConcreteComponent (references CIM_ProcessorCore and CIM_Processor)	Optional	See section 10.3.
CIM_ElementCapabilities (references CIM_HardwareThread and CIM_EnabledLogicalElementCapabilities)	Optional	See section 10.4.
CIM_ElementCapabilities (references CIM_Memory and CIM_EnabledLogicalElementCapabilities)	Optional	See section 10.5.
CIM_ElementCapabilities (references CIM_Processor and CIM_ProcessorCapabilities)	Optional	See section 10.6.
CIM_ElementCapabilities (references CIM_ProcessorCore and CIM_EnabledLogicalElementCapabilities)	Optional	See section 10.7.
CIM_EnabledLogicalElementCapabilities	Optional	See sections 7.6.1, 7.7.1, 7.8.1, and 10.7.
CIM_HardwareThread	Optional	See section 10.9.

Element Name	Requirement	Description
CIM_Memory	Optional	See sections 10.10 and 7.8.
CIM_Processor	Mandatory	See sections 7.1 and 10.11.
CIM_ProcessorCapabilities	Optional	See sections 7.2 and 10.12.
CIM_ProcessorCore	Optional	See section 10.13.
CIM_RegisteredProfile	Mandatory	See section 10.14.
CIM_SystemDevice	Mandatory	See section 10.15.
Indications		
None defined in this profile		

10.1 CIM_AssociatedCacheMemory

CIM_AssociatedCacheMemory associates an instance of CIM_Processor or CIM_ProcessorCore with an instance of CIM_Memory that represents the cache memory of the processor. Table 30 contains the requirements for elements of this class.

Table 30 – Class: CIM_AssociatedCacheMemory

Elements	Requirement	Notes
Antecedent	Mandatory	**Key:** This property shall reference the instance of CIM_Memory that represents the cache memory.
Dependent	Mandatory	**Key:** This property shall reference the instance of CIM_Processor or CIM_ProcessorCore. See section 7.8 for more details.
Level	Mandatory	None
WritePolicy	Mandatory	None
CacheType	Mandatory	None
ReadPolicy	Mandatory	None
Associativity	Mandatory	None
OtherLevelDescription	Conditional	This property shall be implemented when the Level property has a value of 1 (Other).
OtherWritePolicyDescription	Conditional	This property shall be implemented when the WritePolicy property has a value of 1 (Other).
OtherCacheTypeDescription	Conditional	This property shall be implemented when the CacheType property has a value of 1 (Other).

10.2 CIM_ConcreteComponent—References CIM_HardwareThread and CIM_ProcessorCore

CIM_ConcreteComponent associates an instance of CIM_ProcessorCore (the Threading Processor Core) with an instance CIM_HardwareThread that represents a hardware thread. CIM_ConcreteComponent shall be instantiated when the Threading Processor Core and the instance of CIM_HardwareThread are instantiated. Table 31 contains the requirements for elements of this class.

Table 31 – Class: CIM_ConcreteComponent—References CIM_HardwareThread and CIM_ProcessorCore

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** This property shall reference the Threading Processor Core.
PartComponent	Mandatory	**Key:** This property shall reference the CIM_HardwareThread that represents the hardware thread.

10.3 CIM_ConcreteComponent—References CIM_ProcessorCore and CIM_Processor

CIM_ConcreteComponent associates an instance of CIM_Processor (the Host Processor) with an instance CIM_ProcessorCore that represents a processor core. CIM_ConcreteComponent shall be instantiated when the Host Processor and the instance of CIM_ProcessorCore are instantiated. Table 32 contains the requirements for elements of this class.

Table 32 – Class: CIM_ConcreteComponent—References CIM_ProcessorCore and CIM_Processor

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** This property shall reference the Host Processor.
PartComponent	Mandatory	**Key:** This property shall reference the CIM_ProcessorCore that represents the hosted processor cores.

10.4 CIM_ElementCapabilities—References CIM_HardwareThread and CIM_EnabledLogicalElementCapabilities

CIM_ElementCapabilities associates an instance of CIM_HardwareThread with the instance of CIM_EnabledLogicalElementCapabilities that describes the capabilities of the instance of CIM_HardwareThread.

CIM_ElementCapabilities is mandatory when the instance of CIM_HardwareThread and the instance of CIM_EnabledLogicalElementCapabilities that describes the capabilities of the instance of CIM_HardwareThread exist. Table 33 contains the requirements for elements of this class.

Table 33 – Class: CIM_ElementCapabilities—References CIM_HardwareThread and CIM_EnabledLogicalElementCapabilities

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** This property shall reference the instance of CIM_HardwareThread.
Capabilities	Mandatory	**Key:** This property shall reference the instance of CIM_EnabledLogicalElementCapabilities.

10.5 CIM_ElementCapabilities—References CIM_Memory and CIM_EnabledLogicalElementCapabilities

CIM_ElementCapabilities associates an instance of CIM_Memory with the instance of CIM_EnabledLogicalElementCapabilities that describes the capabilities of the instance of CIM_Memory.

CIM_ElementCapabilities is mandatory when the instance of CIM_Memory and the instance of CIM_EnabledLogicalElementCapabilities that describes the capabilities of the instance of CIM_Memory exist. Table 34 contains the requirements for elements of this class.

Table 34 – Class: CIM_ElementCapabilities—References CIM_Memory and CIM_EnabledLogicalElementCapabilities

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** This property shall reference the instance of CIM_Memory.
Capabilities	Mandatory	**Key:** This property shall reference the instance of CIM_EnabledLogicalElementCapabilities.

10.6 CIM_ElementCapabilities—References CIM_Processor and CIM_ProcessorCapabilities

CIM_ElementCapabilities associates an instance of CIM_Processor with the instance of CIM_ProcessorCapabilities that describes the capabilities of the instance of CIM_Processor.

CIM_ElementCapabilities is mandatory when the instance of CIM_Processor and the instance of CIM_ProcessorCapabilities exist. Table 35 contains the requirements for elements of this class.

Table 35 – Class: CIM_ElementCapabilities—References CIM_Processor and CIM_ProcessorCapabilities

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** This property shall reference the instance of CIM_Processor.
Capabilities	Mandatory	**Key:** This property shall reference the instance of CIM_ProcessorCapabilities.

10.7 CIM_ElementCapabilities—References CIM_ProcessorCore and CIM_EnabledLogicalElementCapabilities

CIM_ElementCapabilities associates an instance of CIM_ProcessorCore with the instance of CIM_EnabledLogicalElementCapabilities that describes the capabilities of the instance of CIM_ProcessorCore.

CIM_ElementCapabilities is mandatory when the instance of CIM_ProcessorCore and the instance of CIM_EnabledLogicalElementCapabilities that describes the capabilities of the instance of CIM_ProcessorCore exist. Table 36 contains the requirements for elements of this class.

Table 36 – Class: CIM_ElementCapabilities—References CIM_ProcessorCore and CIM_EnabledLogicalElementCapabilities

Elements	Requirement	Notes
ManagedElement	Mandatory	**Key:** This property shall reference the instance of CIM_ProcessorCore.
Capabilities	Mandatory	**Key:** This property shall reference the instance of CIM_EnabledLogicalElementCapabilities.

10.8 CIM_EnabledLogicalElementCapabilities

CIM_EnabledLogicalElementCapabilities represents the capabilities of the memory, the processor core, or the hardware thread. Table 37 contains the requirements for elements of this class.

Table 37 – Class: CIM_EnabledLogicalElementCapabilities

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
RequestedStatesSupported	Mandatory	See sections 7.6.1.1, 7.7.1.1, and 7.8.1.1.
ElementNameEditSupported	Mandatory	See sections 7.6.1.2, 7.7.1.2, and 7.8.1.1.
MaxElementNameLen	Conditional	See sections 7.6.1.3, 7.7.1.3, and 7.8.1.3.

10.9 CIM_HardwareThread

CIM_HardwareThread represents the hardware thread of the processor. Table 38 contains the requirements for elements of this class.

Table 38 – Class: CIM_HardwareThread

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
LoadPercentage	Optional	EXPERIMENTAL
EnabledState	Mandatory	See section 7.7.4.
RequestedState	Mandatory	See section 7.7.3.
OperationalStatus	Mandatory	None
HealthState	Mandatory	None
ElementName	Mandatory	The property shall match the pattern ".*".
RequestStateChange( )	Conditional	See section 8.3.

10.10 CIM_Memory

CIM_Memory represents the CPU's cache memory. Table 39 contains the requirements for elements of this class.

Table 39 – Class: CIM_Memory

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	**Key**
CreationClassName	Mandatory	**Key**
SystemName	Mandatory	**Key**
DeviceID	Mandatory	**Key**
BlockSize	Mandatory	None
NumberOfBlocks	Mandatory	None
EnabledState	Mandatory	See section 7.8.4.
RequestedState	Mandatory	See section 7.8.3.
HealthState	Mandatory	None
OperationalStatus	Mandatory	None
ElementName	Mandatory	The property shall match the pattern ".*".
RequestStateChange( )	Conditional	See section 8.4.

10.11 CIM_Processor

CIM_Processor represents the processor or CPU. Table 40 contains the requirements for elements of this class.

Table 40 – Class: CIM_Processor

Elements	Requirement	Notes
SystemCreationClassName	Mandatory	**Key**
SystemName	Mandatory	**Key**
CreationClassName	Mandatory	**Key**
DeviceID	Mandatory	**Key**
Family	Mandatory	None
CurrentClockSpeed	Mandatory	When the EnabledState property has a value of 2 (Enabled), a value of 0 shall indicate that the property value is unknown. When the EnabledState property has a value of 3 (Disabled), this property shall have no meaning.
MaxClockSpeed	Mandatory	When the EnabledState property has a value of 2 (Enabled), a value of 0 shall indicate that the property value is unknown. When the EnabledState property has a value of 3 (Disabled), this property shall have no meaning.
ExternalBusClockSpeed	Mandatory	When the EnabledState property has a value of 2 (Enabled), a value of 0 shall indicate that the property value is unknown. When the EnabledState property has a value of 3 (Disabled), this property shall have no meaning.
CPUStatus	Mandatory	See section 7.5.1.
LoadPercentage	Optional	None
EnabledState	Mandatory	See section 7.5.2.
RequestedState	Mandatory	See section 7.4.
OperationalStatus	Mandatory	None
HealthState	Mandatory	None
ElementName	Mandatory	The property shall match the pattern “.*”.
OtherFamilyDescription	Conditional	This property shall be implemented if the Family property contains the value “Other”.
RequestStateChange( )	Conditional	See section 8.1.

10.12 CIM_ProcessorCapabilities

CIM_ProcessorCapabilities represents the capabilities of the processor. Table 41 contains the requirements for elements of this class.

Table 41 – Class: CIM_ProcessorCapabilities

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
NumberOfProcessorCores	Mandatory	A value of 0 shall mean “Unknown”.
NumberOfHardwareThreads	Mandatory	A value of 0 shall mean “Unknown”.
RequestedStatesSupported	Mandatory	See section 7.2.2.
ElementNameEditSupported	Mandatory	See section 7.2.4.
MaxElementNameLen	Conditional	See section 7.2.5.

10.13 CIM_ProcessorCore

CIM_ProcessorCore represents the core of the processor. Table 42 contains the requirements for elements of this class.

Table 42 – Class: CIM_ProcessorCore

Elements	Requirement	Notes
InstanceID	Mandatory	**Key**
CoreEnabledState	Mandatory	See section 7.6.4.1.
LoadPercentage	Optional	EXPERIMENTAL
EnabledState	Mandatory	See section 7.6.4.2.
RequestedState	Mandatory	See section 7.6.3.
OperationalStatus	Mandatory	None
HealthState	Mandatory	None
ElementName	Mandatory	The property shall match the pattern ".*".
RequestStateChange( )	Conditional	See section 8.2.

10.14 CIM_RegisteredProfile

The CIM_RegisteredProfile class is defined by the *Profile Registration Profile*. The requirements denoted in Table 43 are in addition to those mandated by the *Profile Registration Profile*.

Table 43 – Class: CIM_RegisteredProfile

Elements	Requirement	Notes
RegisteredName	Mandatory	This property shall have a value of "*CPU*".
RegisteredVersion	Mandatory	This property shall have a value of "1.0.0".
RegisteredOrganization	Mandatory	This property shall have a value of 2 (DMTF).

NOTE: Previous versions of this document included the suffix "Profile" for the RegisteredName value. If implementations querying for the RegisteredName value find the suffix "Profile", they should ignore the suffix, with any surrounding white spaces, before any comparison is done with the value as specified in this document.

10.15 CIM_SystemDevice

CIM_SystemDevice associates an instance of CIM_Processor with the instance of CIM_ComputerSystem of which the CIM_Processor instance is a member. Table 44 contains the requirements for elements of this class.

Table 44 – Class: CIM_SystemDevice

Elements	Requirement	Notes
GroupComponent	Mandatory	**Key:** This property shall reference the instance of CIM_ComputerSystem of which the instance of CIM_Processor is a member.
PartComponent	Mandatory	**Key:** This property shall reference the instance of CIM_Processor.

ANNEX A
(informative)

Change Log

Version	Date	Description
1.0.0c	2006/07/02	Preliminary Version of the Profile
1.0.0	2008-10-31	Final Version of the Profile

ANNEX B
(informative)

Acknowledgments

The authors wish to acknowledge the following people.

Editors:

- Jon Hass – Dell
- Khachatur Papanyan – Dell
- Jeff Hilland – HP

Contributors:

- Jon Hass – Dell
- Khachatur Papanyan – Dell
- Jeff Hilland – HP
- Christina Shaw – HP
- Aaron Merkin – IBM
- Jeff Lynch – IBM
- Perry Vincent – Intel
- John Leung – Intel