Management Component Transport Protocol (MCTP) Host Interface Specification

Information for Work-in-Progress version:

IMPORTANT: This document is not a standard. It does not necessarily reflect the views of DMTF or its members. Because this document is a Work in Progress, this document may still change, perhaps profoundly and without notice. This document is available for public review and comment until superseded.

Provide any comments through the DMTF Feedback Portal: https://www.dmtf.org/standards/feedback

CONTENTS

1 Foreword

The *Management Component Transport Protocol (MCTP) Host Interface Specification* (DSP0256) was prepared by the PMCI Working Group.

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. For information about the DMTF, see http://www.dmtf.org.

1.1 Acknowledgments

The DMTF acknowledges the following individuals for their contributions to this document:

Editors:

- Hemal Shah — Broadcom Inc.

- Philip Chidester — Dell Inc.

- Samer El-Haj-Mahmoud — Arm Limited

Contributors:

- Darren Hart — Ampere Computing

- Harb AbdulHameed — Ampere Computing

- Robert Gough — Ampere Computing, Intel Corporation

- Thanu Rangarajan — Arm Limited

- Jose Marinho — Arm Limited

2 Introduction

The Management Component Transport Protocol (MCTP) defines a communication model intended to facilitate communication between:

- Management controllers and other management controllers

- Management controllers and management devices

The communication model includes a message format, transport description, message exchange patterns, and configuration and initialization messages. The *MCTP Base Specification* (DSP0236) describes the protocol and commands used for communication within and initialization of an MCTP network

This *MCTP Host Interface Specification* (DSP0256) describes how MCTP host interfaces are discovered (interfaces between a management controller and host software).

2.1 Document conventions

The conventions described in the following clauses apply to this specification.

2.2 Reserved and Unassigned Values

Unless otherwise specified, any reserved, unspecified, or unassigned values in enumerations or other numeric ranges are reserved for future definition by the DMTF.

Unless otherwise specified, numeric or bit fields that are designated as reserved shall be written as 0 (zero) and ignored when read.

2.3 Byte Ordering

Unless otherwise specified, byte ordering of multi-byte numeric fields or bit fields is "Big Endian" (that is, the lower byte offset holds the most significant byte, and higher offsets hold lesser significant bytes).

Management Component Transport Protocol (MCTP) Host Interface Specification

3 Scope

This document provides the specifications for the Management Component Transport Protocol (MCTP) host interface.

4 Normative References

The following referenced documents are indispensable for the application of this document. For dated or versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies. For references without a date or version, the latest published edition of the referenced document (including any corrigenda or DMTF update versions) applies.

DMTF DSP0236, *Management Component Transport Protocol (MCTP) Base Specification* https://www.dmtf.org/standards/published_documents/DSP0236_1.3.x.pdf

DMTF DSP0239, *Management Component Transport Protocol (MCTP) IDs and Codes Specification* https://www.dmtf.org/standards/published_documents/DSP0239_1.9.x.pdf

DMTF DSP0233, *Management Component Transport Protocol (MCTP) I3C Transport Binding Specification* https://www.dmtf.org/standards/published_documents/DSP0233_1.0.x.pdf

DMTF DSP0237, *Management Component Transport Protocol (MCTP) SMBUS/I2C Transport Binding Specification* https://www.dmtf.org/standards/published_documents/DSP0237_1.2.x.pdf

DMTF DSP0238, *Management Component Transport Protocol (MCTP) PCIe VDM Transport Binding Specification* https://www.dmtf.org/standards/published_documents/DSP0238_1.2.x.pdf

DMTF DSP0254, *Management Component Transport Protocol (MCTP) KCS Transport Binding Specification* https://www.dmtf.org/standards/published_documents/DSP0254_1.0.x.pdf

DMTF DSP0284, *Management Component Transport Protocol (MCTP) Memory-Mapped BMC Interface (MMBI) Transport Binding Specification* https://www.dmtf.org/standards/published_documents/DSP0284_1.0.x.pdf

DMTF DSP0292, *Management Component Transport Protocol (MCTP) PCC Transport Binding Specification 1.0* https://www.dmtf.org/sites/default/files/standards/documents/DSP0292_1.0.X.pdf

UEFI Forum, *Advanced Configuration and Power Interface Specification (ACPI)* https://uefi.org/specifications

IPMI Consortium*, Intelligent Platform Management Interface Specification*, v2.0 Revision 1.1 October 1, 2013 https://www.intel.com/content/www/us/en/products/docs/servers/ipmi/ipmi-second-gen-interface-spec-v2-rev1-1.html

DMTF DSP0134, *System Management BIOS (SMBIOS) Reference Specification* https://www.dmtf.org/dsp/DSP0134

5 Terms and Definitions

Refer to DSP0236 for terms and definitions that are used across the MCTP specifications. For the purposes of this document, the following terms and definitions apply.

6 Symbols and Abbreviated Terms

Refer to DSP0236 for symbols and abbreviated terms that are used across the MCTP specifications. For the purposes of this document, the following additional symbols and abbreviated terms apply.

7 MCTP Host Interface

The MCTP Host Interface defines how MCTP packets are delivered over a host interface. This specification describes the host interface discovery and commands for registering software endpoints such as BIOS, UEFI or system software.

7.1 Host Interface Discovery
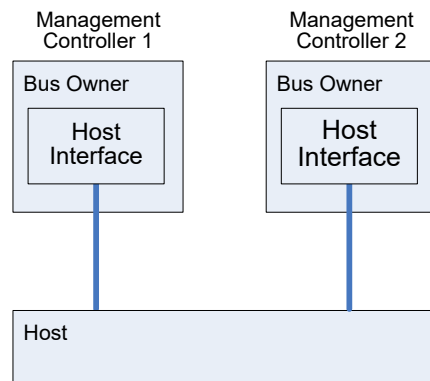
Figure 1 — MCTP Host Interface Discovery

A Management Controller implements a single host interface. There can be multiple Management Controllers in a system, but each Management Controller is in a separate MCTP network that connects to the Host.

Each MCTP host interface can be discovered using ACPI or additionally using SMBIOS Type 42 tables.

7.2 Multiple Host Interfaces

Multiple host interfaces may exist in a given system implementation. Refer to the "MCTP Overview" clause in DSP0236 for details on identifying whether multiple host interfaces connect to common or separate MCTP networks.

A client, of a management controller, is permitted to interact with different Host Interfaces in order to identify the Host Interface that connects to its management controller. The client can use the "Get Endpoint UUID" MCTP command to identify the management controller on each Host Interface.

7.3 Transport-Specific Commands

For correct Host Interface operation, the MCTP Host Interface may require transport-specific commands to be implemented. The details of these command are described in the transport-binding specifications.

8 Locating MCTP Host Interfaces with ACPI

This specification introduces the option of describing the presence of the MCTP Host Interface using ACPI namespace.

8.1 Locating MCTP Host Interfaces in ACPI Name Space

MCTP Host interfaces should be described in the ACPI name space. This makes it possible for the OSPM (an ACPI term for "OS Operating System-directed configuration and Power Management" essentially meaning an ACPI-aware OS or OS loader) to enumerate the MCTP Host interface as a device.

Note that to be ACPI compatible, the fixed resources for MCTP Host interfaces must still be accounted for in accordance with the ACPI Specification. If the device is not formally described in the ACPI Name Space, its resources must be described as fixed system resources or the resources appended to some other fixed resource system device in order to ensure that the OS does not attempt to allocate those resources to some other device.

To formally describe the MCTP Host interface in ACPI Name Space, an MCTP device is created using the named device object. The MCTP device object can have the elements as described in Table 1.

Table 1 — MCTP Device Object Control Methods

Object	Description	Support Level
_ADR	Named object that evaluates to the interface's address on its parent bus. _ADR is a standard device configuration control method defined in the ACPI Specification. Note: A device object must contain either an _HID object or an _ADR object, but must not contain both.	Required only for devices on a bus that has standard enumeration mechanism.
_HID	Named object that provides the interface's Plug and Play identifier. This value can be vendor specific but must set to DMT0001[1] if no CID object is provided. _HID is a standard device configuration control method defined in the ACPI Specification. Note: A device object must contain either an _HID object or an _ADR object, but must not contain both.	Required only for devices that do not use standard bus enumeration mechanism.
_CID	Named object that provides the interface's compatible Plug and Play identifier. This object is required and contains the value of DMT0001 if _HID is present and contains vendor specific identifier. Otherwise, this object is optional.	See the "Description" column for this object
_STR	Named object that evaluates to a Unicode string that may be used by an OS to provide information to an end user describing the device. _STR is a standard device configuration control method defined in the ACPI Specification. The strings "MCTP_KCS", "MCTP_Serial", "MCTP_I2C", "MCTP_I3C", "MCTP_PCIE", "MCTP_MMBI", "MCTP_PCC" are recommended for identifying the KCS, Serial, I^2C/SMBus, I3C, PCIe VDM, MMBI MCTP, and PCC host interfaces, respectively.	Required

_UID	Named object that specifies a device's unique persistent ID, or a control method that generates it. _UID is a standard device configuration control method defined in the ACPI Specification. The value 0 must be used to identify the primary MCTP Host Interface.	Required if more than one device is present.
_CRS	Named object that returns the interface's current resource settings. System Processor Management Interfaces are considered static resources; hence only return their defined resources. The address region definition is interface type/subtype dependent. _CRS is a standard device configuration control method defined in the ACPI Specification.	Recommended
_STA	Object that returns the status of the device: enabled, disabled or removed, as defined in the ACPI Specification. If this method is not present, the device is assumed to be enabled.	Recommended
_IFT	Object that specifies the MCTP Host interface type, as defined in DSP0239. The OS driver uses the _IFT object to determine the link-layer type. (Note: _IFT has been reserved in ACPI Specification as the control method name defined by the SPMI table in the IPMI Specification and by this specification.)	Required
_SRV	Object that specifies the MCTP Base specification revision. (Note: _SRV has been reserved in ACPI Specification as the control method name defined by the SPMI table in the IPMI Specification and by this specification.)	Required

1. DMTF has registered the DMT PNP ID with https://uefi.org/PNP_ACPI_Registry for describing all DMTF related devices. DMTF has granted the use of DMT0001 to describe the generic Management Controller Device as defined in this specification.

In a multi-node system where there may be more than one MCTP device in an OS domain, it is highly recommended that all MCTP devices be described in the ACPI name space with the _STA returning enabled for the active MCTP device(s).

8.2 Locating MCTP Host Interfaces via SMBIOS tables

The SMBIOS DSP0134 specification defines the Management Controller Host Interface table (with type set to 42). The SMBIOS type 42 table describes the interface and protocol types. For each Protocol type, there is a Protocol Type Specific Data entry.

8.2.1 Protocol Type Specific Data

For an MCTP HI, the SMBIOS Type 42 has the Interface Type set to a value from the "MCTP Host Interfaces" range. The values in the "MCTP Host Interfaces" range are defined in DSP0239, in the "MCTP host interface type identifiers" Section. There must be at least one Protocol Record entry. That mandatory Protocol Record must have Protocol Type set to MCTP (03h). The Protocol Type Specific Data for the Protocol Record with 03h Protocol Type is defined in Table 2.

Table 2 — Protocol Type Specific Data

Offset	Field Name	Size	Description

00h	MCTP protocol version	DWORD	The version of the MCTP base protocol that the management controller implements.
04h	MCTP link-layer type	DWORD	Integer describing the type of link or protocol layer that connects the Host to the management controller. The value is as defined in the MCTP host interface type identifier table from DSP0239.
08h	Instance number	DWORD	The instance number of the MCTP host interface. The instance number must match with the _UID method of one ACPI DSDT device with _HID=DMT0001. Note: at least one instance must exist
0Ch	Characteristics	DWORD	Reserved, must be 0.

8.2.2 Interface Type Specific Data

The Interface Type Specific data is defined for the MMBI interface type. This data is reserved for the remaining interface types, occupying the minimum required size of 4 bytes, as specified by this field definition defined by Table 3 which is a DSP0134 type 42 structure.

Table 3 — Interface Type Specific Data

Interface type	Interface Type Specific Data definition			
	Offset	Field Name	Size	Description
MMBI	00h	MMBI Capability Descriptor pointer	QWORD	A pointer to the MMBI capability descriptor, as defined in DSP0282.
	Offset	Field Name	Size	Description
I2C, I3C, KCS	00h	Reserved	DWORD	Reserved, must be zero.

8.3 Example MCTP Definition ASL Code

Example ASL code that defines MCTP Host interfaces is provided in the following subclauses.

Multiple MCTP Host interfaces can exist. There must be a DSDT ACPI table for each interface. Each such table has a unique _UID value. The _UID value for this device must be an integer. One of the tables must have _UID=0. The instance number set is sparse, e.g. a Instance number set of {0, 2, 10, …} representing a collection of devices with _UID=0, _UID=2 and _UID=10, … is legal.

8.3.1 Example 1: KCS Interface in 64-bit Address Space

Example ASL for describing a memory-mapped KCS host interface, located in a 64-bit address space at address 0x80000FFFFC020CA2. For more details, refer to the MCTP KCS Binding specification.

```
Device(MHI0)
{
  Name(_HID, "DMT0001")           // MCTP host interface
  Name(_STR, Unicode("MCTP_KCS")) // String to identify MCTP host interface
  Name(_UID, 0)    // UID for the primary MCTP host interface in the system

  // Returns the "Current Resources"
  Name(_CRS,
  ResourceTemplate() {
    QWordMemory(
    ResourceConsumer, //
    PosDecode, //
    MinFixed, //
    MaxFixed, //
    NonCacheable, //
    ReadWrite, //
      0xFFFFFFFFFFFFFFFF,   // \_GRA, Address granularity.
                           // E.g. All 64-bits decoded.
      0x80000FFFFC020CA2,   // \_MIN, Address range minimum
                           // (System I/F base addr.)
      0x80000FFFFC020CA4,   // \_MAX, Address range max
      0x0000000000000000,   // \_TRA, Translation.
                           // 0 for non-bridge devices
      0x0000000000000002,   // \_LEN, Address range length
         ,                  // Resource Source Index
         ,                  // Resource Source Name
         ,                  // A name to refer back to this resource
         ,                  // \_MTP, Nothing=\>AddressRangeMemory
         ,                  // \_TTP, Translation. Nothing=\>TypeStatic
                           // TypeTranslation: This resource, which is memory
                           // on the secondary side of the bridge is I/O on
                           // the primary side of the bridge.
                           // TypeStatic: This resource, which is memory on
                           // the secondary side of the bridge is also memory
                           // on the primary side of the bridge.
      )
    }
    )

  // Returns the interface type
  Method(_IFT) {
    Return(0x02) // MCTP KCS (DSP0239, Table 4)
  }
```

```
    // Returns the interface specification revision
    Method(_SRV) {
      Return(0x0130) // MCTP Base Specification Revision 1.3
    }

    // This interface does not support interrupt


}
```

8.3.2 Example 2: I^{2}C/SMBus Interface

Example ASL for describing the presence of the MCTP Host Interface using I^{2}C/SMBus MCTP binding. In order to associate the MCTP Host Interface with a particular SMBus host controller interface, the SMBus host controller must be described as a resource under the MCTP HI device DSDT table. For Example:

```
Device(SMB0) // example SMBus host controller
{
  Name(_HID, "<Vendor-Specific HID>") // Vendor-specific HID
  Name(_UID, 0) // Unique ID of particular host controller
  :
  :
} // end Device SMB0

Device (MHI0)
{
Name(_HID, "DMT0001") // MCTP host interface
Name(_STR, Unicode("MCTP_I2C")) // String to identify MCTP host interface
Name(_UID, 0) // UID for the primary MCTP host interface in the system

// Current Resource Settings of the SMBus interface
Name(_CRS, ResourceTemplate() {
  I2CSerialBusV2(
    0x0010,                // I2C bus target address
    ControllerInitiated,   // Target mode
    400000,                // Maximum connection speed
    AddressingMode7Bit,    // Addressing mode
    "\\_SB.SMB0",          // Resource source (I2C bus controller)
    0x00,                  // Resource Source index
    ResourceConsumer,      // Resource usage
    MCTP_HI0,              // Descriptor name
    Exclusive,             // Shared or Exclusive
                           // Vendor specific data

  )
}

Method (_STA, 0, NotSerialized) {
```

```
  Return (0x0F)
}


// Returns the interface type
Method(_IFT) {
  Return (0x09) // MCTP I2C (DSP0239, Table 4)
}

// Returns the interface specification version
Method(_SRV) {
  Return(0x0130) // MCTP Base Specification Revision 1.3
}


} // end Device MHI0
```

8.3.3 Example 3: I3C Interface

Example ASL for describing the presence of the MCTP Host Interface using I3C MCTP binding. In order to associate the MCTP Host Interface with a particular I3C device, the I3C device must be described as a resource under ACPI and the MCTP Host Interface defined as a device under the host controller. For example:

```
Device(I3C0) // example I3C host controller
{
  Name(_HID, "MIPI0100") // MIPI defined standard I3C HCI (https://www.mipi.org/hid)
  Name(_UID, 0) // Unique ID of particular host controller
  :
  :
  Device (MHI0)
  {
    ~~Name(_HID, "DMT0001") // MCTP host interface~~
    Name(_STR, Unicode("MCTP_I3C")) // String to identify MCTP host interface
    Name(_UID, 0) // UID for the primary MCTP host interface in the system

    Method (_ADR, 0, NotSerialized)
      Return (0x??) // TODO. See comment…
    }

    Method (_STA, 0, NotSerialized) {
      Return (0x0F)
    }

    // Returns the interface type
    Method(_IFT) {
      Return (0x0A) // MCTP I3C (DSP0239, Table 4)
    }

    // Returns the interface specification version
```

```
        Method(_SRV) {
          Return(0x0130) // MCTP Base Specification Revision 1.3
        }

      } // end Device MHI0
    } // end Device I3C0
```

8.3.4 Example 4: MMBI Interface

Example ASL for describing the presence of the MCTP Host Interface using MMBI MCTP binding.

A MCTP HI device, using MMBI, points to the based address of the MMBI shared buffer. The device must return 0xC from the _IFT method.

An example of a MMBI MCTP HI ASL description is:

```
Device(MHI0)
{
  Name(_HID, "DMT0001") // MCTP host interface
  Name(_STR, Unicode("MCTP_MMBI")) // String to identify MCTP host interface
  Name(_UID, 0)    // UID for the primary MCTP host interface in the system

  // Returns the "Current Resources"
  Name(_CRS,
  ResourceTemplate() {
    QWordMemory(
    ResourceConsumer, //
    PosDecode, //
    MinFixed, //
    MaxFixed, //
    NonCacheable, //
    ReadWrite, //
      0xFFFFFFFFFFFFFFFF,   // \_GRA, Address granularity.
                           // E.g. All 64-bits decoded.
      0x80000FFFFC020CA2,   // \_MIN, Address range minimum
                           // (System I/F base addr.)
      0x80000FFFFC020CA4,   // \_MAX, Address range max
      0x0000000000000000,   // \_TRA, Translation.
                           // 0 for non-bridge devices
      0x0000000000001000,   // \_LEN, Address range length
          ,                 // Resource Source Index
          ,                 // Resource Source Name
          ,                 // A name to refer back to this resource
          ,                 // \_MTP, Nothing=\>AddressRangeMemory
          ,                 // \_TTP, Translation. Nothing=\>TypeStatic
                           // TypeTranslation: This resource, which is memory
                           // on the secondary side of the bridge is I/O on
                           // the primary side of the bridge.
```

```
                                    // TypeStatic: This resource, which is memory on
                                    // the secondary side of the bridge is also memory
                                    // on the primary side of the bridge.
                )
        }

        // Returns the interface type
        Method _IFT {
                Return (0x0C) // MCTP MMBI (DSP0239, Table 4)
        }

        // Returns the interface specification version
        Method(_SRV) {
                Return(0x0130) // MCTP Base Specification Revision 1.3
        }
    }
```

8.3.5 Example 4: PCC Interface

Example ASL for describing the presence of the MCTP Host Interface using PCC MCTP binding.

A MCTP HI device, using PCC, contains the PCC space identifiers, of both the type 3 and type 4 PCC channel, in a _ CRS object.. The device must return 0x0D from the _IFT method.

An example of a PCC MCTP HI ASL description is:

```
Device(PCC0)
{
  Name(_HID, "DMT0001") // MCTP host interface
  Name(_STR, Unicode("MCTP_PCC")) // String to identify MCTP host interface
  Name(_UID, 0)    // UID for the primary MCTP host interface in the system

  // Returns the "Current Resources"
  Name(_CRS,
  ResourceTemplate() {
    WordPCC (
      1, // PccChannel, the index of the PCC subspace in the PCCT
      , // ResourceSourceIndex
      , // ResourceSource
      , // DescriptorName
    ),
    WordPCC (
      2, // PccChannel, the index of the PCC subspace in the PCCT
      , // ResourceSourceIndex
      , // ResourceSource
      , // DescriptorName
    )
  }
```

```
        // Returns the interface type
    Method _IFT {
            Return (0x0D) // MCTP PCC (DSP0239, Table 4)
    }

        // Returns the interface specification version
    Method(_SRV) {
            Return(0x0130) // MCTP Base Specification Revision 1.3
    }
}
```

Annex A

(informative)

Notation

9 ANNEX A (informative) Notation

Examples of notations used in this document are as follows:

- 2:N In field descriptions, this will typically be used to represent a range of byte offsets starting from byte two and continuing to and including byte N. The lowest offset is on the left, the highest is on the right.

- (6) Parentheses around a single number can be used in message field descriptions to indicate a byte field that may be present or absent.

- (3:6) Parentheses around a field consisting of a range of bytes indicates the entire range may be present or absent. The lowest offset is on the left, the highest is on the right.

- PCIe Underlined, blue text is typically used to indicate a reference to a document or specification called out in clause 2 or to items hyperlinked within the document.

- rsvd Abbreviation for "reserved." Case insensitive.

- [4] Square brackets around a number are typically used to indicate a bit offset. Bit offsets are given as zero-based values (that is, the least significant bit [LSb] offset = 0).

- [7:5] A range of bit offsets. The most significant bit is on the left, the least significant bit is on the right.

- 1b The lower case "b" following a number consisting of 0s and 1s is used to indicate the number is being given in binary format.

- 0x12 A leading "0x" is used to indicate a number given in hexadecimal format.

10 ANNEX B (informative) Change Log

Version	Date	Description
1.0.0	2010-07-21	Released as DMTF Standard
2.0.0	2024-03-04	Fixed document title Added acknowledgments list Updated references versions and links Removed the SMBIOS, PCI/PCIe, and static ACPI MCHI table methods for MCTP Host Interface discovery Updated the ACPI namespace discovery method to cover newer MCTP bindings (including I^2C/SMBus and I3C) Updated ACPI ASL examples to add I^2C/SMBus and I3C