MCTP 2.0 Overview v0.2

PMCI September 14th 2021

Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change without notice. The standard specifications remain the normative reference for all information.
- For additional information, see the DMTF website.
- This information is a summary of the information that will appear in the specifications. See the specifications for further details.

Introduction

- MCTP specs were first released in 2009. Since then, the specifications have become widely adopted. Enabling MCTP use on current and future platforms requires addressing some key developments:
  - The number of managed devices in a single platform has increased dramatically
  - Message sizes have increased significantly in some use cases
  - The number of message types that are being transferred over MCTP has grown
  - Security & encryption are now a requirement in most environments
  - The transports for MCTP have increased beyond the original small packet interfaces
- In order to keep MCTP as the management transport protocol of choice in the coming decade, the industry has reached the point where these issues should be addressed. And yet, this needs to be done carefully to leverage the success of MCTP 1.0, which will remain in some environments for years to come.

Overview

- MCTP 2.0 is intended to address some of the deficiencies and limitations of MCTP 1.0
- MCTP 2.0 may be introduced in incremental steps. This document defines the content for each of the new capabilities which are planned to be included in MCTP 2.0

General Requirement

- MCTP 2.0 allows us to start from scratch. However, there is a lot of good things in existing MCTP 1.0 base. The request is to leverage as much as possible the functionality that exists in MCTP 1.0.
  - This whole slide set assumes that a lot of MCTP 1.0-base feature and function carries forward into 2.0 in some shape or form (e.g. discovery, bridging, commands, MTU discovery, etc). It does not call out every feature and function in 1.0

Planned Features list

Feature #	Feature name
1	Extending EID range
2	Enabling coexistence of MCTP 1.0/2.0
4	Hot-Insertion/removal support
5	Backward compliance with existing HW
6a	Increasing concurrent messages count
6b	Increase the number of outstanding packets
9	Forward looking header format
10	Medium and protocol agnostic header
11	Negotiate transmission unit size
13	Support for Secured Messages
14	Provide support for reliable & efficient large transfers
16a	Keep Rate-based Flow control support
17	Reliably detect message corruption

Feature #1 - Extending EID range

- Problem:
  - Scalability beyond 255 EIDs
  - Large MCTP network can easily consume 255 endpoint IDs.

- Requirement:
  - Add support for up to 64K
  - Increase the endpoint IDs namespace or whatever the equivalent of endpoint ID is for MCTP 2.0
- Comment
  - Current MCTP binding specifications may have to be adapted to accommodate EIDs beyond 255

Feature #2 - Enabling coexistence of MCTP 1.0/2.0

- MCTP 1.0 and MCTP 2.0 shall be allowed to be used on the same physical bus
- MCTP 1.0 and MCTP 2.0 shall be allowed to be used on the same MCTP network
- MCTP 2.0 compliant bridge shall be able to route both MCTP 1.0 and MCTP 2.0 packets without being required to perform translations

Feature #4 - Hot-Insertion/removal support

- Add new method for propagating the hot-add/remove info up in the hierarchy of bus owners up to the topmost bus owner.
  - Information about endpoints added/removed on a local bus is not visible to other devices, except the local bus owner assigning the EID.
  - Having current information about MCTP endpoints is important for devices that manage the MCTP network, such as a BMC.
    The request is to define such a capability.
- Add new method for endpoints to subscribe for async notifications for Hot-Insertion/removal events
  - Need to address security hazards and considerations with such subscription

Feature #5 - Backward compliance with existing HW

- Any change for MCTP packet header must
  - Add method for discovery of MCTP version support by bus owner and endpoint
  - Allow any endpoint to publish its MCTP version support
  - Allow co-existence of MCTP 1.X and MCTP 2.X messages on the same physical bus
    - Mandatory (shall)
      - 2.X Bus owners and management controllers shall be able to send and receive messages using both formats
      - 2.X Bridges shall support both MCTP 2.X and MCTP 1.X
    - Recommended (should)
      - Require MCTP 2.X capable devices to be able to send and receive messages using both formats
      - Designate EIDs such that there will be no conflict in EIDs at system-level
        - Example: assume all MSBs of EIDs of MCTP 1.X to be all 0's
  - Allow MCTP 2.X capable devices to communicate with MCTP 1.X only devices

Feature #6a - Increasing concurrent messages count

- Allow increased usage of MCTP via:
  - Increase the number of tags (4-bit)

Feature #6b - Increase the number of outstanding packets

- Allow increased usage of MCTP via:
  - Increase the number of outstanding packets - more bits for sequence Number

Feature #9 Forward looking header format

- Problem:
  - It is hard to accommodate changes to MCTP header for future request and issues that we have to do a new 2.0 base.
    - Example, we have to do a 2.0 just to increase the endpoint namespace size.
- Requirement:
  - Allow for future changes to MCTP base or header without requiring a 3.0 spec change.
    - Example, decouple the “Hdr version” from the physical binding/requirements

Feature #10 Medium and protocol agnostic header

- Problem:
  - It is hard to transfer an MCTP message over various mediums especially when they are bridged.
    - This is because of the “Hdr version” having language that binds it to the physical transport
  - Also, it is not clear in a mixed environment how various bridges and endpoints of different MCTP protocol can work together.
- Requirement:
  - Make it clear and easy that an MCTP message and its header can go across various mediums and endpoints with various MCTP versions without changing the MCTP message and MCTP packet header contents.
    - Example: the MCTP header should not have any bindings to physical layer.

Feature #11 Negotiate transmission unit size

- Problem:
  - The minimum packet size is too restrictive for present day use. Thus, lots of overhead at the packet level (i.e. many packets need to be sent).
- Requirement:
  - Enable negotiating the transmission unit size (while retaining the default BTU)
  - Negotiating the Transmission Unit Size (shall be >= BTU) and be medium specific across all message types applied for both directions
  - Negotiation of transmission unit size should **only** be part of MCTP base
  - Support negotiating MCTP packet size between two MCTP Endpoints
- Comments
  - How to address bridged connections?
    - Can we make the negotiation End2End?
    - Should it include the bus-owner in the negotiation?
    - Should we support message-type specific MSMTU negotiation?

Feature #13 Support for secured messages

- Provide secured message capability integrated directly into MCTP 2.0 without requirement for separate message type
  - ie MCTP Type 6 no longer would be required
    - MCTP Type-6 is mutex with MCTP native secured messages
- Comment
  - Definition of Message Integrity within MCTP base
  - Support of native secured messages shall be optional
  - MCTP Bridges shall be secured message agnostic
    - MCTP bridging and routing function shall be secured messages agnostic
  - Negotiation for support for secured messages shall be end-to-end

Feature #14 Provide support for reliable & efficient multi-packet message transfers

- Problem
  - If a packet is dropped or damaged in multi-packet MCTP message, MCTP relies on the higher level to retry. This means the entire message needs to be retried. This can reduce reliability in the transport and cause additional overhead that consumes bandwidth.
  - Error criteria shall be within the list of causes to drop a packet
- Requirement:
  - Support reliable message delivery between two MCTP endpoints
    - Allowing a destination receiver to request a retransmission of MCTP packet(s)
    - Allowing a source transmitter to support of retransmission of MCTP packet(s)
    - Handling an error shall be part of the same message
  - Enable negotiating the max message size that can use re-transmission/re-assembly across all message types between two MCTP endpoints

Feature #16a Keep Rate-based Flow control support

- A Negotiated sender-based rate-limiting feature:
  - Rate-based flow control
- Transport-Level only capability
- Decided that flow-control is
  - End-to-end feature

Feature #17 Reliably detect message corruption

- Problem:
  - Need a common method for MCTP to reliably detect message corruption
- Requirement:
  - Define a mechanism in MCTPv2 Base (optional but recommended) to use integrity check for fragmented messages - simple CRC or SPDM could be the options
- Motivation:
  - Packet loss detection – improve incorrect message reassembly detection
    - MCTPv1 does not define any mechanism (other than the seqNo/TAG field, which is very weak) to detect message payload corruption due to packet loss – see next slide
    - Higher-layer protocols **assume** that MCTP layer delivers uncorrupted messages, for example, the CERTIFICATE message in SPDM spec, which will likely be fragmented into many fragments, does not have any integrity check
  - Efficiency – integrity (signature of the payload) is a common need nowadays
    - side-effect of such a signature check is corruption detection so the proposal is to reuse it for efficiency reasons