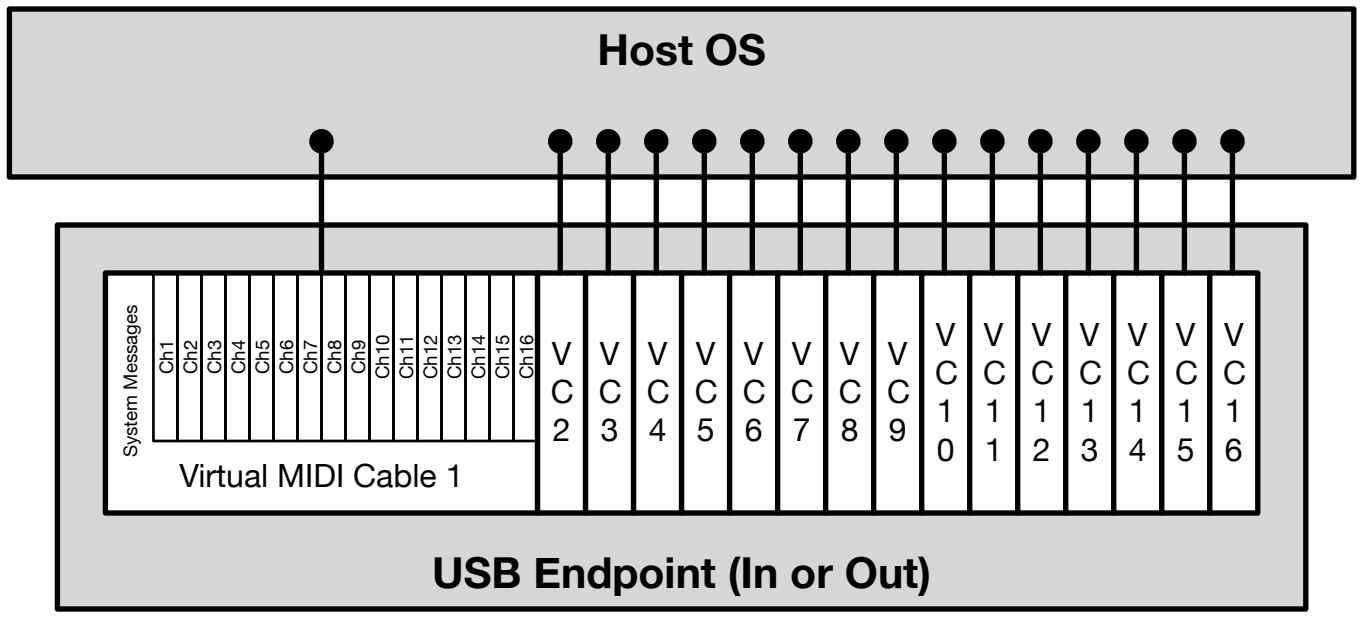
MIDI 2.0 USB Device - Topology, Connections, and Addressing

Mike Kent Draft 1 October 12, 2022

MIDI 1.0



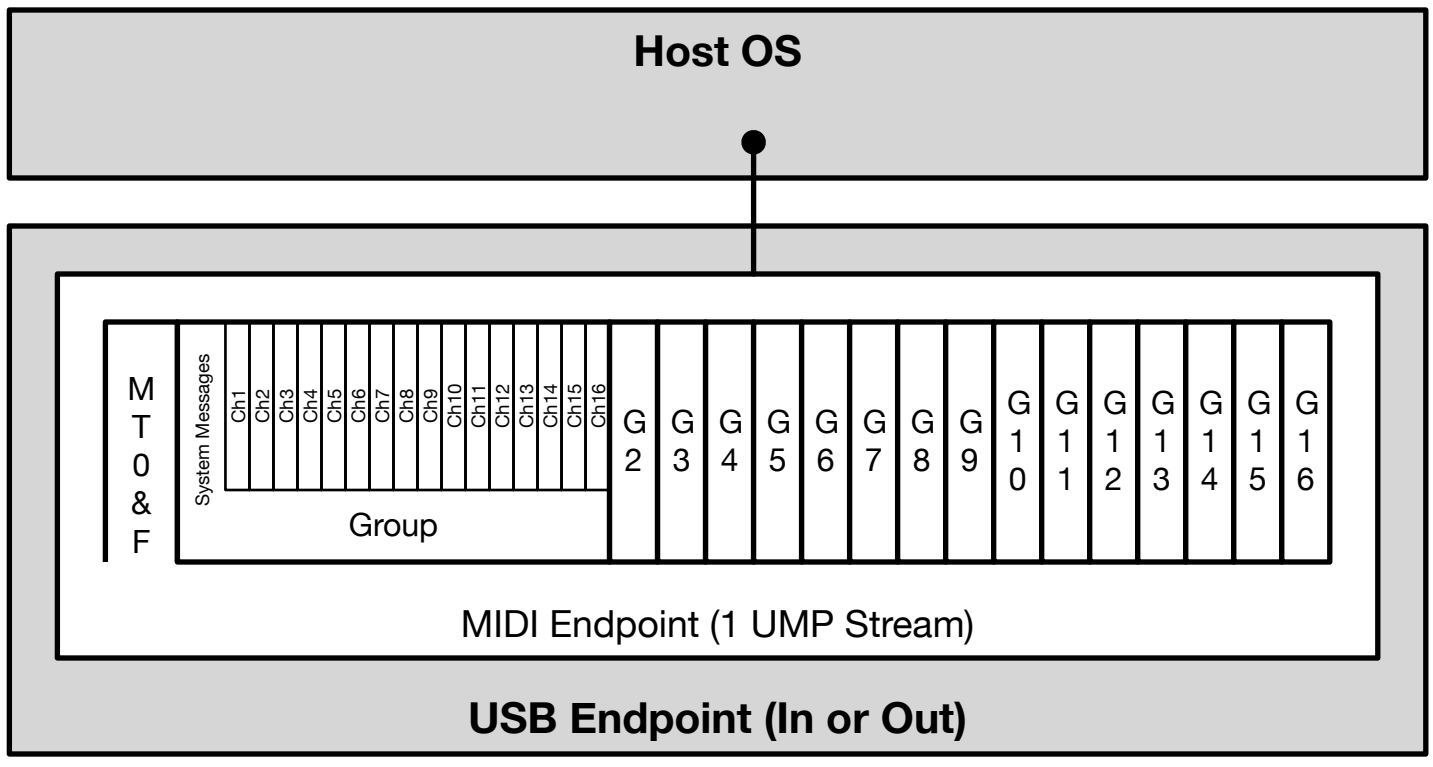
API exposes MIDI Ports as Connection Pins.
Each MIDI Port represents a single Virtual MIDI Cable.

Up to 16 Virtual MIDI Cables declared in USB Descriptors

Applications determine addressing via fields in the messages:
1. 16 MIDI Channels in the Cable
2. System Messages apply across the whole Cable

Most Devices have 1 Input USB Endpoint and 1 Output USB Endpoint

MIDI 2.0



API exposes MIDI Endpoints as Connection Pins.
Each MIDI Endpoint represents a single UMP Stream.

16 Groups. Active Groups are declared in USB Descriptors (Group Terminal Blocks*). Active Groups also declared by Function Block discovery messages at the application layer.

Applications determine addressing via fields in the messages:
Routing to each of 16 Groups:
1. 16 MIDI Channels in the Group
2. System Messages apply across the whole Group**

Message Type 0x0 and 0xF:
Apply across the whole MIDI Endpoint

Most Devices have 1 Input USB Endpoint and 1 Output USB Endpoint

MIDI 1.0 Applications in MIDI 2.0 Environment

A Group is roughly equivalent to a MIDI 1.0 Virtual Cable for backward compatibility purposes

A MIDI 1.0 Application does not know about Groups. Potential strategies:
- API needs to set addressing to map messages to/from a user-selected Active Group.

or perhaps
- API represents Active Groups as MIDI 1.0 Ports to MIDI 1.0 Applications. API needs to Set addressing to map messages to/from the Group which is represented by the MIDI Port

or there may be other possible solutions

In either case:
- Data Format translation is necessary at the connection pin
- Protocol translation is sometimes necessary at the connection pin

** Group Terminal Blocks have problematic limitations in latest changes to MIDI 2.0 architecture. Function Blocks are intended to replace Group Terminal Blocks.*

*** System Messages apply across a Group by default but if Function Blocks are known, then System Messages apply across the Function Block. These diagrams do not show Function Blocks.*