This presentation covers Gen-Z Operation Classes, also referred to as OpClasses.
Disclaimer

This document is provided 'as is' with no warranties whatsoever, including any warranty of merchantability, noninfringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. Gen-Z Consortium disclaims all liability for infringement of proprietary rights, relating to use of information in this document. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted herein.

Gen-Z is a trademark or registered trademark of the Gen-Z Consortium.
All other product names are trademarks, registered trademarks, or servicemarks of their respective owners.

All material is subject to change at any time at the discretion of the Gen-Z Consortium

http://genzconsortium.org/
Gen-Z organizes operations into OpClasses. Each OpClass contains 32 OpCodees, where each OpCode represents a unique operation, e.g., a Read or Write operation.

Gen-Z supports a total of 34 OpClasses. Three implicit OpClasses and 32 explicit OpClasses.

Implicit OpClasses are configured on a per component interface basis and are intended for use in point-to-point, mesh, and daisy-chain topologies. Implicit OpClass packets do not contain an OpClass Label field.

Explicit OpClasses are intended for use in point-to-point and switch-based topologies. Explicit OpClass packets contain an OpClass Label field that explicitly identifies the OpClass.
This slide illustrates a generic explicit OpClass packet. There are two fields of interest: the OpCode field identifies the specific operation type and the OpClass Label (OCL) field identifies the specific OpClass. Each is a 5-bit field respectively providing 32 OpCodes and 32 Explicit OpClasses.
The following three implicit OpClasses are specified: P2P 64 and P2P-Vendor-defined.

- The P2P 64 OpClass is optimized for point-to-point and mesh topologies between any type of component.
- The P2P-Vendor-defined OpClass is optimized for point-to-point solutions using a vendor-defined protocol.

The following explicit OpClasses are specified:

- Core 64—Single or multi-subnet with 64-bit effective addressing
- Control—in-band management operations, event notification, etc.
- Atomic 1—Atomic operations
- Large Data Move 1—Buffer operations and large Read operations
- Advanced 1—Pattern / Regular Expression operations and Lightweight Notification operations
- Advanced 2—Precision Time, Unicast Packet Encapsulation, etc.
- Context ID—Used for operations that target a Responder context identifier (handle)
- Multicast—Unreliable and Reliable Multicast operations
- Strong Ordered Domain (SOD)—strong ordering operations
- 8 Vendor Defined OpClasses—Vendor specified or de facto operations

Vendor-defined operations will be covered in the next slide.
In addition to these standard OpClasses, Gen-Z supports 8 vendor-defined OpClasses. These OpClass may be used to communicate vendor-specific operations or de facto standard operations. A UUID is used to uniquely identify each OpClass. This enables the architecture to support nearly any number of vendor-defined operations and, by using UUIDs, to significantly simplify management and enable vendors to innovate and collaborate without requiring a central management authority.

The UUID is not present on the wire since that would be extremely inefficient. Instead, it is mapped to an OpClass Label field. Hence, any component can support up to 8 different vendor-defined OpClasses at a given time, and these are transparently transported across the Gen-Z topology.
OpClasses are managed through the OpCode Set structure. Two fields are provisioned per OpClass: the supported OpCode set and the enabled OpCode Set. As the names imply, the Supported field indicates which OpCodes are supported and the Enabled field indicates which OpCodes a component may use. Further, each field is composed of sub-fields to indicate each OpCode’s role, i.e., whether it supports and is enabled to use an operation as a Requester, as a Responder, or as a Requester-Responder.

A component may support multiple OpCode Set structures to enable interoperability with a variety of disparate components or across multiple technology generations.
Thank you

This concludes this presentation. Thank you.