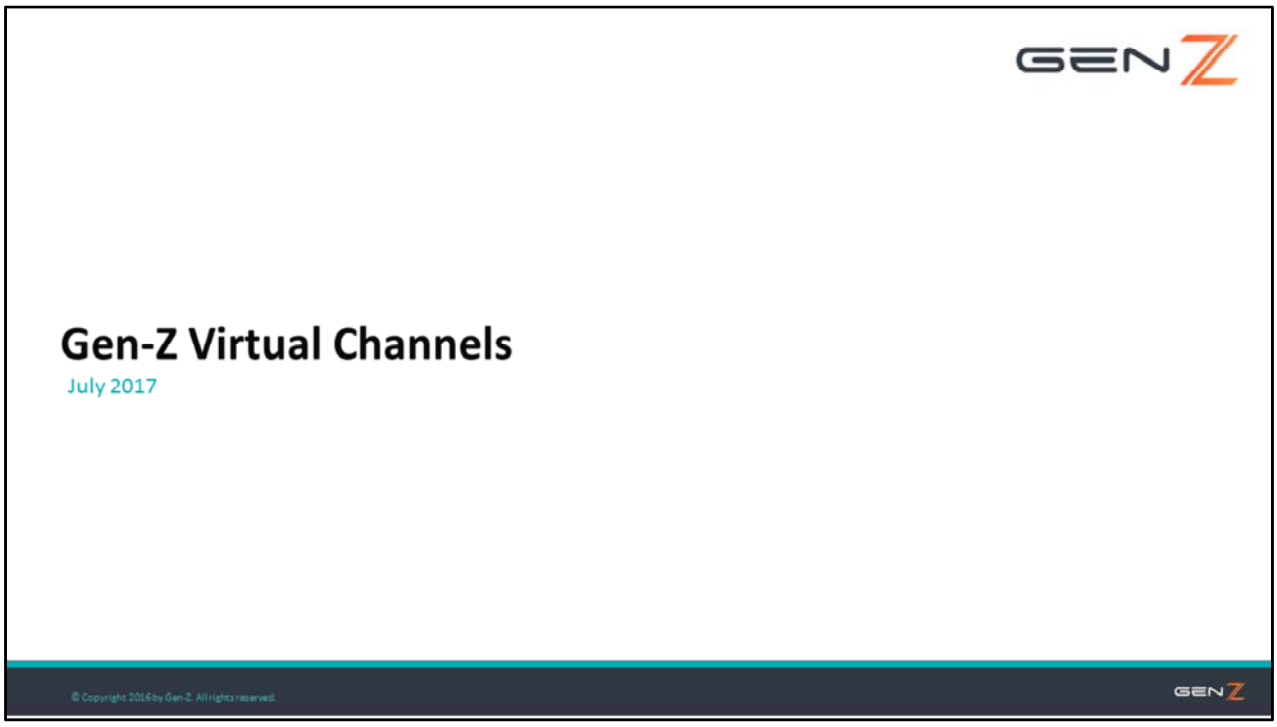


This presentation covers Gen-Z's use of Virtual Channels including how Traffic Classes are used as an input into VC selection.

Disclaimer

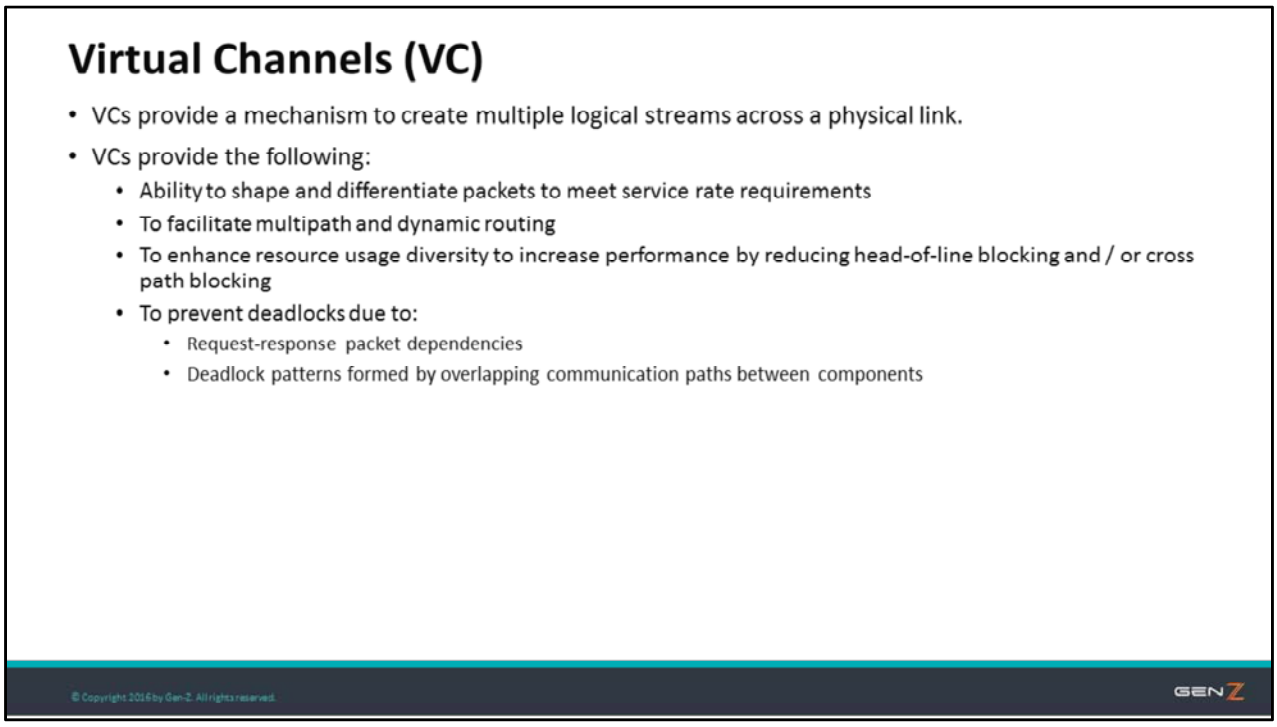
This document is provided 'as is' with no warranties whatsoever, including any warranty of merchantability, noninfringement, fitness for any particular purpose, or any warranty otherwise arising out of any proposal, specification, or sample. Gen-Z Consortium disclaims all liability for infringement of 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/

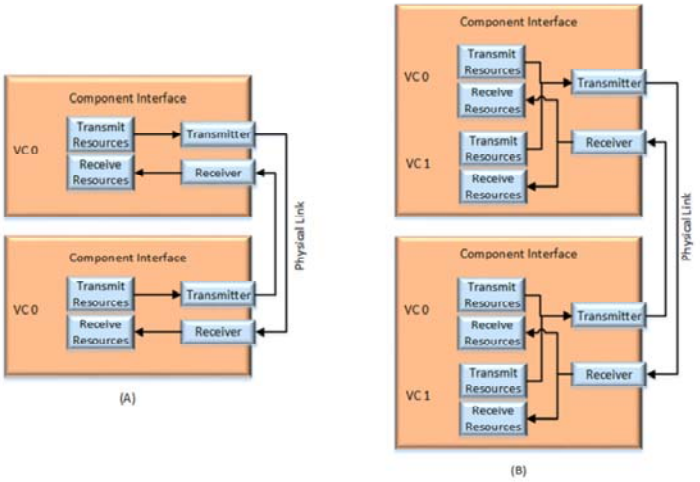


A virtual channel (VC) is a mechanism used to create multiple logical communication streams across a physical link. VCs are used to:

- Shape and differentiate packets to meet specific service rate requirements. VC arbitration is used to determine how frequently a VC is serviced.
- To facilitate multipath and dynamic routing. The VC acts as an input into path selection, e.g., VCs associated with higher priority traffic take the left path and lower priority VCs take the right path.
- To enhance resource usage diversity, VCs can be dynamically remapped, e.g., the VC within a packet injected by a component interface with fewer VCs can be remapped in a component that supports many VCs to spread traffic out
- To prevent deadlocks, e.g., request-response packet dependencies and deadlock patterns formed when packets cross specific paths between components
- Not discussed in this presentation, VCs can be configured to support PCIe Compatible Ordering (PCO). This is used by Logical PCI Devices that require communications be restricted to a single path and single VC to preserve PCIe ordering

VC Requirements

- Each VC logically represents a set of per-interface, per-direction transmit and receive buffers and tracking logic and resources
- Each VC identifies a unique flow-control domain
- VCs are delineated by identifiers, VC0, VC1, …
- Each interface is required to support VC0
- Each interface may support up to 32 VCs
  - If multiple VCs are supported, then each interface shall support 0 to N consecutive VCs
  - P2P-Coherency interfaces support up to 8 VCs
- Gen-Z VCs are typically much lighter-weight than VCs used in some alternative technologies, e.g., a PCIe VC represents 3 resource sets compared to Gen-Z's 1 resource set



Each VC represents a set of transmit and receive buffers at each interface on a link. Each VC identifies the set of flow-control credits used to manage these buffers and prevent buffer overflow.

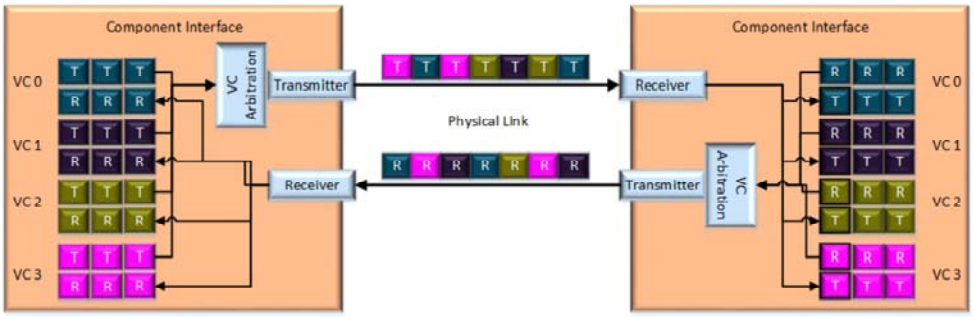
VCs are identified by an integer, 0-31.

All interfaces are to support VC0, and any additional VCs are required to be monotonically increasing in value, i.e., 0, 1, 2, 3, …

An interface can support any number of VCs up to the maximum supported by a given OpClass.

- Component interfaces that support P2P-Core, support a single implicit VC, VC0.
- Component interfaces that support P2P-Coherency, support up to 8 VCs—VC0-VC7
- Component interfaces that support explicit OpClass packets, support up to 32 VCs—VC0-VC31

Link Time Multiplexing



- End-to-end packets on different VCs may be interleaved across a physical link
- If two interfaces support disparate number of VCs, then the lowest common set of VCs is enabled
  - For example, if interface A supports 4 VCs and interface B supports 2, then VC0 and VC1 are enabled
- VC arbitration is outside of the specification's scope
  - Interfaces ensure all VCs are periodically serviced to ensure packets make forward progress

Packets from different VCs are time-multiplexed across a physical link. This figure illustrates interfaces that support 4 VCs—VC0-VC3. Using a VC arbitration policy that is outside of the specification's scope, each interface transmits packets from each VC if it has available flow-control credits.

If one interface supports fewer VCs than the other, management configures the lowest common denominator, e.g., if one supported 4 VCs and the other 3 VCs, then management would only enable VC0, VC1, and VC2.

An interface that supports only VC0 does not time multiplex packets; everything is exchanged using VC0.

Deadlock Avoidance

- To avoid Request-Response dependencies, request and response packets are sent on different VCs
  - For example, request packets are on VC1 and response packets on VC0
  - Request and response packets cannot be intermixed on the same VC in the same direction, i.e., request packets flow one direction and response packets flow in the opposite direction on a given VC
- Component Destination structure is used to manage VC configuration
  - Requesters select one of N possible paths and one of the available VCs used to transmit request packets
  - Responders select one of N possible paths and one of the available VCs used to transmit response packets

GEN Z

To avoid Request-response dependencies, request and response packets are transmitted on different VCs. Request packets can flow on any VC, and response packets can flow on any different VC.

Within a Requester or a Responder (only those that support explicit OpClasses), the Component Destination structure is used to select an egress interface and one of the available VCs. The structure contains a set of tables that are indexed using the destination component's identifiers and the traffic class (if a request).

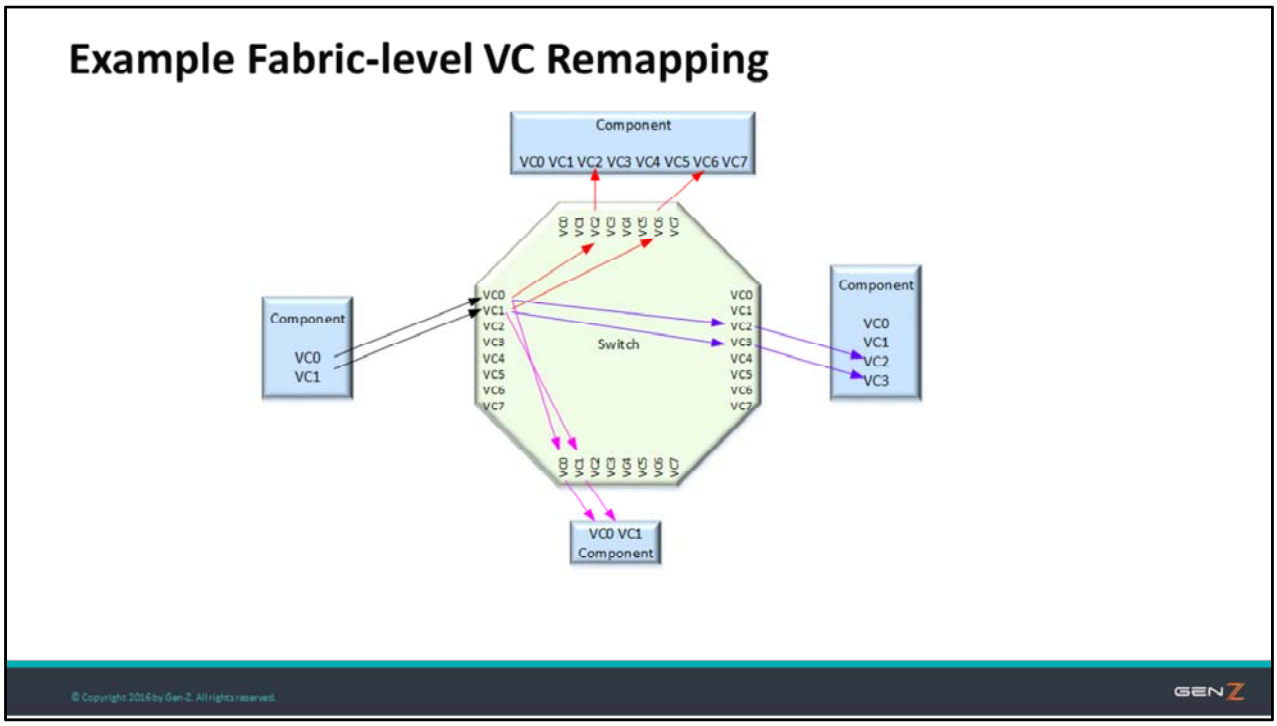
VC Remapping

- Packet relay components such as switches may support VC remapping
  - VC remapping enables the VC field within a packet to be remapped to a different VC value
  - VC remapping enables:
    - A switch to spread traffic from components with a limited number of VCs across multiple VCs within a network
      - For example, traffic from multiple dual-VC components can be spread across switches with up to 32 VCs to reduce congestion and head-of-line blocking latency
      - Within a single egress interface, a switch can dynamically remap traffic from a congested VC to a less or non-congested VC
      - For example, this enables packets to be transmitted on the same physical link but on a less congested VC to enable more VCs and associated resources to be used to mitigate congestion effects
- PCRC and ECRC Impacts
  - PCRC validated prior to performing VC remapping
  - PCRC and ECRC are not recalculated, but are modified to reflect the new VC
    - Errors within the original CRC will be present in the new CRC.

GEN Z

Packet relay components can support VC remapping. VC remapping takes the packet's VC as an input, in conjunction with the egress interface, and selects the same or a different VC. VC remapping provides multiple benefits as illustrated in this slide.

When the VC is modified, the PCRC and ECRC fields are updated to reflect the new VC value. Modification is performed by applying a difference to the current CRC values. Avoiding recalculation ensures that any error already present in the CRC remains, thus ensuring if a transient error occurs within a switch, it will be detected.



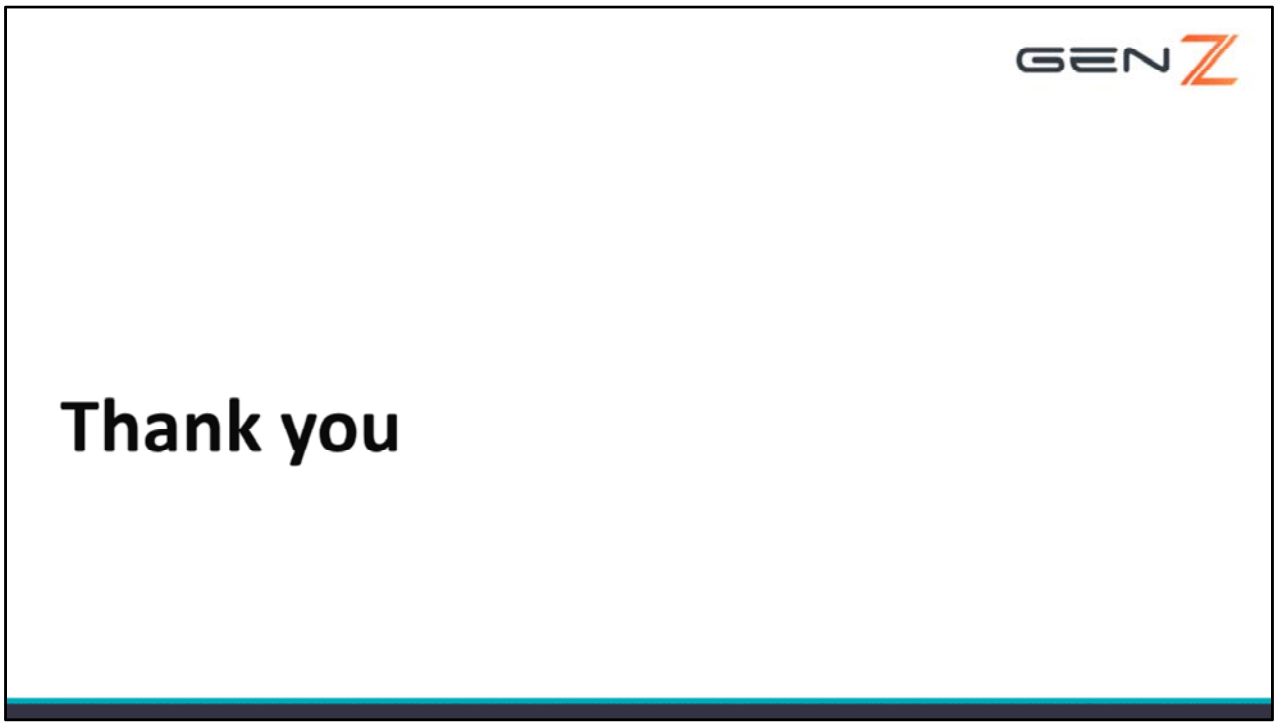
This figure illustrates how VC remapping can be applied to enable components that support different number of VCs to communicate without requiring all components to operate using only the lowest common VCs. For example, two components support only two VCs, one supports four, and one eight. In order to spread resource consumption, the traffic from the two VCs are remapped to different VCs based on the destination component. Though not shown, the same applies when the four VC component communicates with the eight VC component.

Traffic Classes (TC)

- Traffic Class is a 4-bit opaque identifier
  - TC is not explicitly communicated / present within an end-to-end packet, i.e., there is no Traffic Class or priority field within the protocol
  - TC can be configured through a Requester ZMMU or through a component-specific method
- Software maps application QoS or other attributes to a given TC
  - Mapping policies are outside of the specification's scope though may leverage existing application methods
  - For example, IP Diff Serv or 802.1p priority can be mapped to individual TC values
- TC can be used as an input to select an egress interface and VC at the Requester
  - Software associates a VC_{REQ} and a VC_{RSP} with a given TC
  - Responders implicitly comprehend the TC based on the VC_{RSP}
- Switches implicitly comprehend the TC based on the packet's VC field
  - Switches can use the implicit TC to perform VC arbitration and packet relay scheduling
  - Combination of relatively small packet sizes, arbitration, and adaptive scheduling enables a Gen-Z fabric to deliver optimal QoS without requiring packet preemption

GEN Z

Within Requesters / Requester-Responders, a Traffic Class is used as an input into VC selection. For example, if a component supports a Requester ZMMU, then each Requester PTE contains a 4-bit field that indicates the TC. The TC acts as an input into egress interface and VC selection. For example, each TC can be associated with multiple VCs on multiple egress interfaces. TC interpretation and configuration are performed by software, and are outside of the specification's scope.



This concludes this presentation. Thank you.