



SuperSpeed USB Developers Conference

Tokyo, Japan
May 20-21, 2009



Isochronous Protocol

Dan Froelich

USB 3.0 Isochronous Workgroup Chair
Intel Corporation



Agenda

- Background
 - Workgroup Assumptions
 - USB 2.0 Isochronous Review
- USB 3.0 Isochronous Features
- Synchronization
- Isochronous OUT Overview
- Isochronous IN Overview
- Isochronous Framework
 - Service Interval
 - Maximum Packet Size
- Host Isochronous Bursting Flexibility
- Bus Interval Adjustment Mechanism
- Isochronous and Power Management
- Summary

Isochronous Assumptions



- Devices need to be able to synchronize to host time with 200 nanosecond accuracy
 - Generally accepted requirement for high end professional audio ~ 100 ns
- Host should be allowed to provide service anywhere each service interval
 - No strict requirement on order or location of ISO packets, etc.
- No Retries
- Requiring additional device buffering or stream latency is bad if it can be reasonably avoided
- Isoch IN
 - Isochronous IN endpoint does not need to know when packets are dropped
 - Host does need to keep track of when packets are dropped
 - Packets must not be requested too early or too late
- Isoch OUT
 - Host does not need to know when packets are dropped for Isoch OUT
 - Endpoint/device needs to know when packets are dropped
 - Packets must not be sent too early or too late

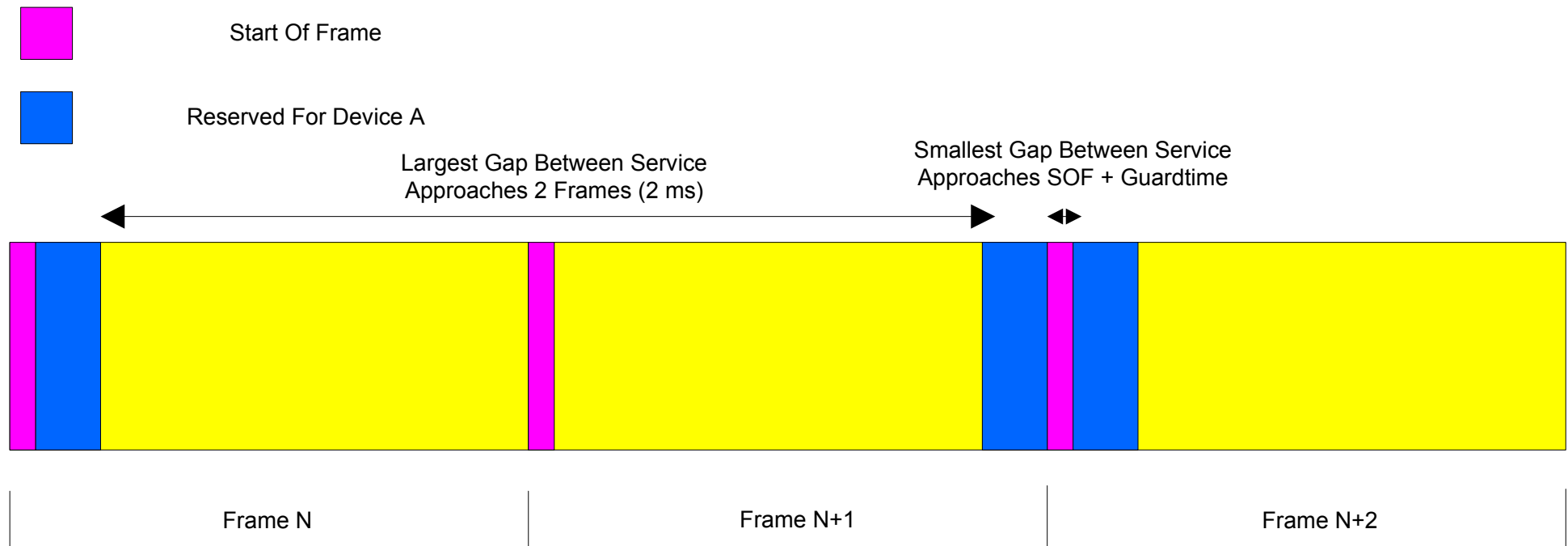
USB 2.0 Isochronous Feature Summary



- Guaranteed Service Attempt Every Service Interval
 - Reserved Bandwidth
 - Service Interval Bound (1 service interval \pm 1 service interval)
 - Typical Service Interval Has Little Jitter
- Reliability
 - Bit Error of 10^{-12} Or Better (PER 10^{-9})
 - No Retries Or Handshaking Needed In Protocol
- Minimal Buffering
 - Typical Implementations Require 2 Service Intervals Of Buffering
- Host Timing Information
 - SOFs Occur At Regular 1 ms/125 μ S Intervals With Small Jitter (10's of Ns)
 - SOFs Can Be Used As Device Clock.
- Data Delivery Time
 - Data Is Sent In Specified Frames/Microframes
 - Data Is Not Too Early or Too Late



Wired USB 2.0 Isochronous Model



- The host may access an endpoint anytime during the service interval
- Isochronous endpoint must not assume any timing relationship between transactions

USB 3.0 Isochronous Goals



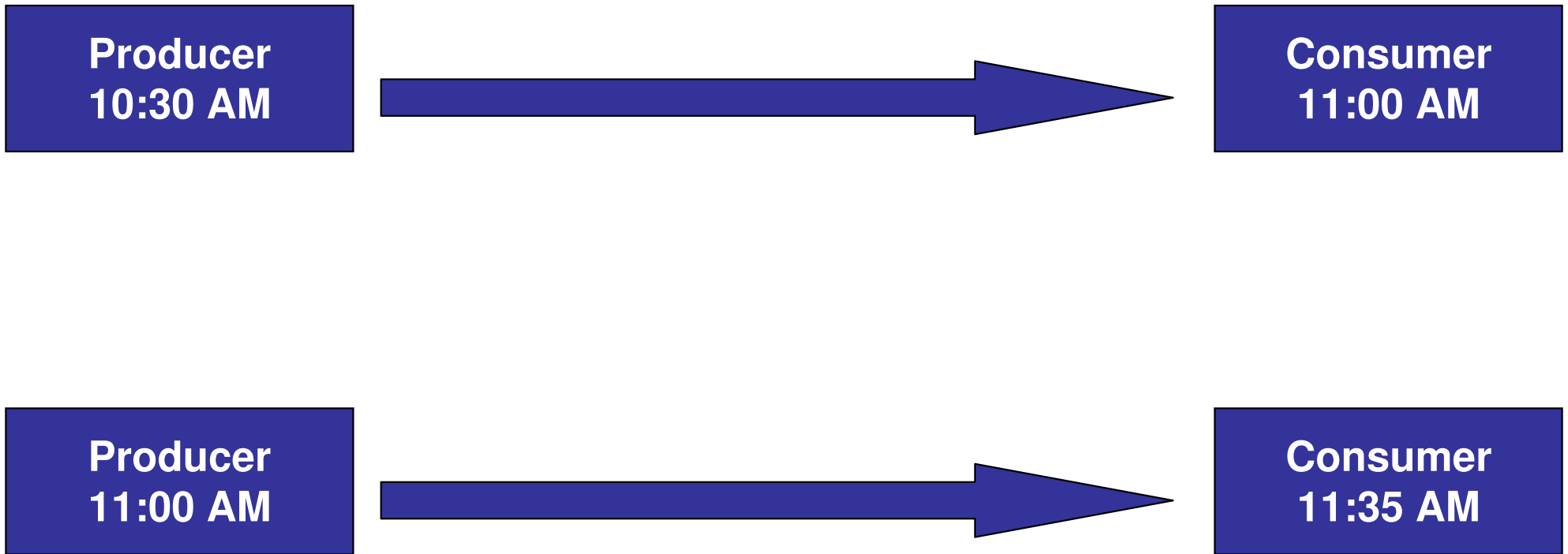
- Preserve Key Characteristics of USB 2.0 Isochronous Protocol
 - Support Applications That Can Tolerate A Small Percentage of Packet Drops and Have Deadlines For Packets
 - Real Time Video
- Improve Power Efficiency of Isochronous Protocol
- Provide Software Backwards Compatibility with Existing Isochronous Drivers

Retries



- No Retries
 - BER target is low ($\sim E-12$ or better)
 - Isochronous protocols of interest can tolerate packet error rates of $E-6$ or better
 - Very simple protocol (no acknowledgements)
 - Must have some other mechanism to make sure you don't get too far ahead (Iso out) or behind (Iso In)
- Retries
 - Can increase effective PER (WUSB)
 - Adds protocol complexity
 - How many retries?
 - When do you retry?
 - If you stop retrying without success – how is it handled?
- **USB 3.0 Isochronous – No Retries**

Synchronization – Problem Statement



Producer And Consumer Have To Have A Way To Set Clocks That Drive Production And Consumption At The Same Rate (Time) Or Buffers Overflow/Underflow

Isochronous Timestamp Packet



- Isochronous Timestamp Packet (ITP) is used to multicast timestamp information from the host to all active devices. ITPs are used to provide host timing information to devices for synchronization.
- ITPs carry no addressing or routing information and are multicast by hubs to all of their downstream ports with links in the U0 state
 - ITP forwarding in Hub does not return a port link to U0
 - ITPs are not forwarded by hub downstream ports not in U0
 - ITPs may be received by non-isochronous devices.
 - **Standard link level processing must still be performed.**
- Host transmits an ITP in every bus interval (125 μ s) within tTimestampWindow (8 μ s) from a bus interval boundary
 - This allows full bus utilization in host transaction scheduling
 - Host does not transmit an ITP if the host port link is not in U0
- Device stay in U0 around bus interval boundary when ITP is needed
- Device may opt not to use received ITP if delayed flag is set
 - Delayed flag is set upon link retry

Fields in Isochronous Timestamp Packet

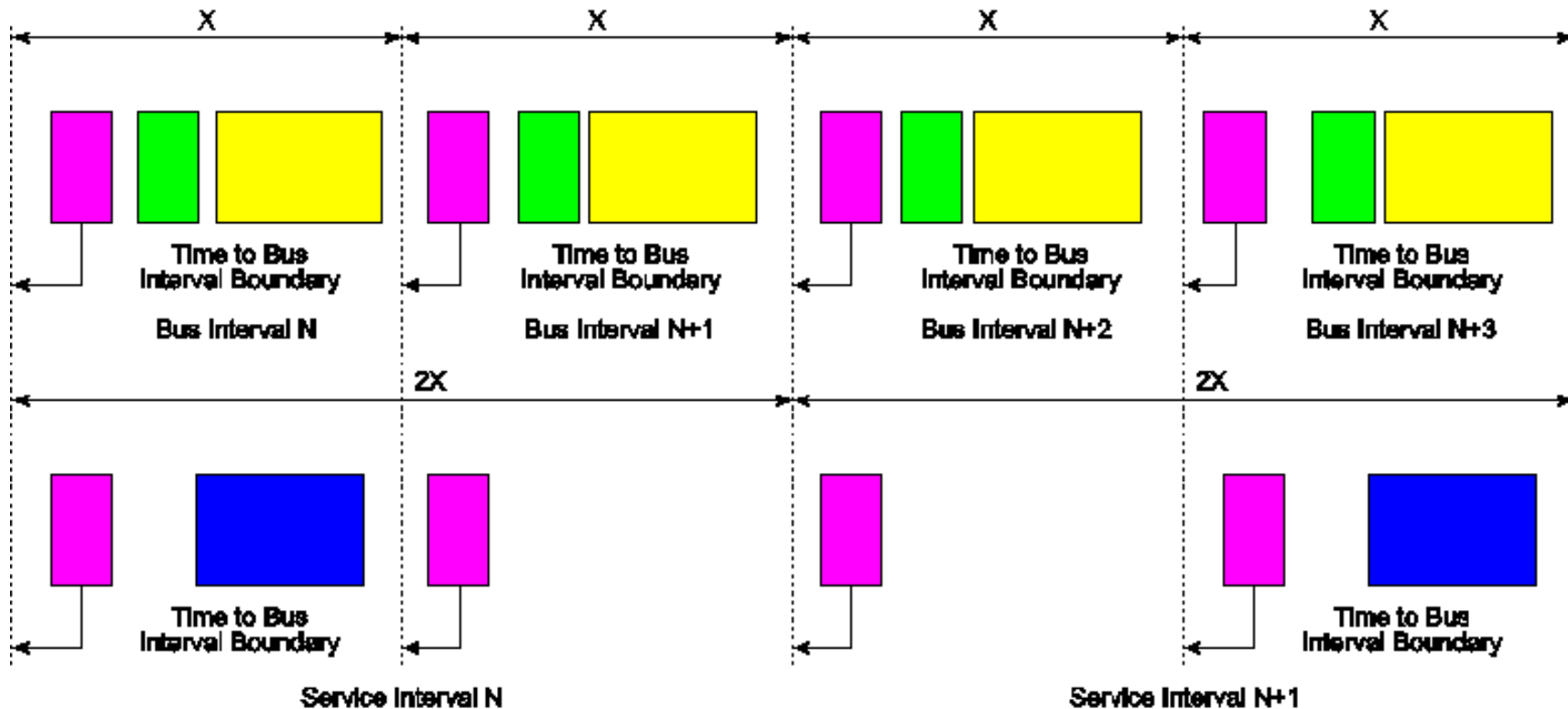


Bus interval counter (14 bits). The 1/8 of a millisecond counter.





Delta (13 bits). The time delta from the start of the current ITP packet to the previous bus interval boundary.

- Timestamp information are generated from non-spread clock
- Timestamps must be accurate to host time at start of ITP transmission +/- 8 HS bit times (16.66 ns)
- Timestamp Fields
 - Bus Interval counter - 1/8 millisecond counter increments every 125 μ s. Counter rollover to 0 when x3FFF is reached
 - Delta – Delta time from the beginning of last bus interval boundary to the beginning of the ITP packet. This is specified in `tlsochTimestampGranularity` (8 HS bit time, 16.666ns)

Service Interval Alignment



Host schedules isochronous transactions with aligned bus interval boundaries across all isochronous endpoints

-  Isochronous Out Header and Data Payload
-  Isochronous Timestamp Packet
-  Isochronous IN Handshake from Host
-  Isochronous IN Header and Data Payload

Synchronization and ITP Propagation Delay

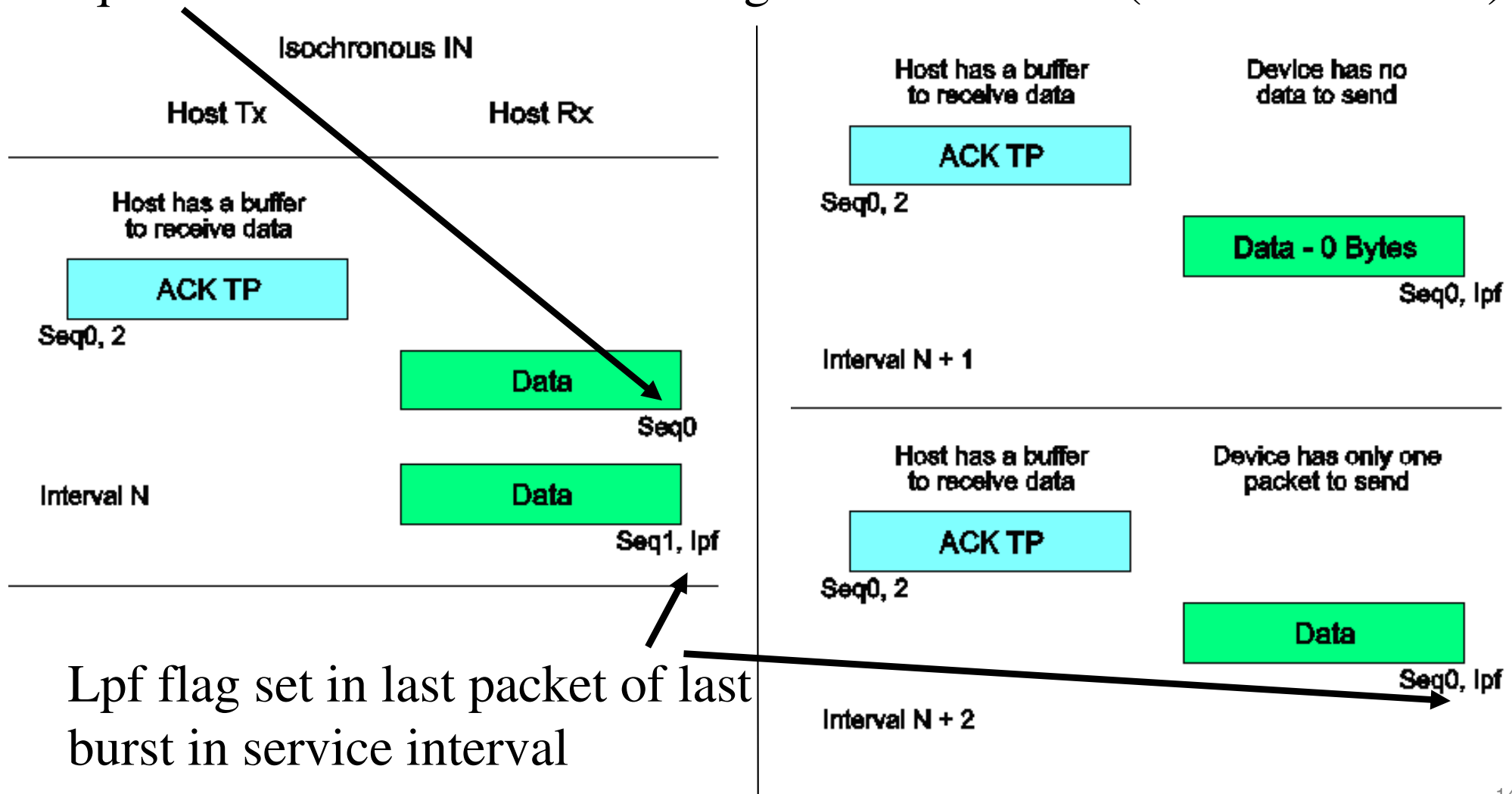


- ITP Propagation Delay
 - Average delay dominated by hub design and number of hubs in hierarchy.
 - Internal clock rate
 - Internal architecture.
 - Average delay reported in descriptors (wHubDelay)
 - Worst case delay variation (200 ns target for worst case)
 - SSC clock mismatches through hierarchy
 - Hub design variation
 - Host ITP accuracy (+/- 8 HS bit time specification limit)
- Accounting for average propagation delay:
 - Delay information (wHubDelay) is reported in each hub. wHubDelay reports the average processing delay in hub when packet is forwarded from upstream port to downstream port.
 - Host system calculates delay in every path and uses Set Isochronous Delay to inform the device the average time delay (0 to 65535ns) starting from a packet transmitted from host to received by device when all links are in U0.
 - Devices uses Isochronous Delay and the timestamp information to know the host time. This allows groups of device to remove average propagation delay from impacting synchronization. For example, HD speakers located in different places of hierarchy



Isochronous IN Sequence

First packet transmitted in service interval always has sequence number 0. Sequence number increases during service interval (rollover after 31)



Isochronous OUT Sequence

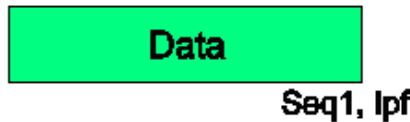


Isochronous OUT

Host Tx

Host Rx

Host has data to send



Host has no data to send

Host has only one data packet to send



First packet transmitted in service interval always has sequence number 0. Sequence number increases during service interval (rollover after 31)

Lpf flag set in last packet of last burst in service interval

Isochronous Endpoint Bandwidth Request



- Service intervals are the same as USB 2.0.
 - $125\mu\text{s} * 2^{(bInterval-1)}$, $bInterval = 1$ to 16 . Possible intervals are $125\mu\text{s}$, $250\mu\text{s}$, $512\mu\text{s}$, $1024\mu\text{s}$, $2048\mu\text{s}$ and up to 4s
- Maximum of 128MB/s for a single burst per service interval
- There can be up to 3 (see mult in bmAttributes of SuperSpeed Endpoint Companion) bursts per service interval – allowing up to 384 MB/s
- Maximum packet size 1KB. All packets transmitted in each service interval must be of maximum packet size, except the last packet
- wMaxPacketSize must be 1024 unless there is only 1 packet per service interval
- Max bandwidth consumed in a service interval is reported in wBytesPerInterval field in endpoint companion descriptor

Host Isochronous Bursting Flexibility



- Host isochronous burst options
 - Transfer all DPs to or from endpoint as single isochronous burst.
 - Split the transfer into smaller bursts of two, four, or eight DPs followed by a final burst for the remaining DPs in the service interval.
 - Sequence number does not reset within service interval
 - LPF flag set only on last packet of last burst within service interval.
- A host may consider these factors when performing Isochronous Bursting:
 - Availability of data buffer
 - Latency of fetching data from/to system memory
 - Volume of data to be transferred to endpoint
 - Device characteristics, reported thru descriptors
 - Non-related transactions that are already scheduled
- Example: The host has 11 packets to send to an isochronous out endpoint during a service interval:
 - 1 burst of 11 packets
 - 1 burst of 8 and 1 burst of 3
 - 1 burst of 4, 1 burst of 4 and 1 burst of 3
 - 5 bursts of 2, and 1 burst of 1
 - No other combinations are allowed

Isochronous Flow Control



- No flow control needed for Isochronous Out
 - Device must provide maximum request buffering anywhere in the service interval, as host may send all data for the interval in a single burst
- No flow control needed for Isochronous IN
 - Host must be able to receive maximum requested data once per interval
 - Device must be able to send all data for the interval in a single burst

Additional USB 3.0 Isochronous Enhancements



- USB 3.0 allows Isochronous capable devices to enter the lower power USB link state between service intervals
 - A SuperSpeed host must transmit a PING packet to the targeted isochronous device before service (if necessary), to transition the path back to the active link state before initiating the isochronous transfer
- USB 3.0 allows small adjustments to 125 μ s service interval
 - A device may change the interval depending on the application's requirement
 - USB 3.0 devices may send a Bus Interval Adjustment Message to the host to adjust its 125 μ s bus interval up to $\pm 13.333\mu$ s

Bus Interval Adjustment Mechanism



- Some devices need to synchronize to an external clock (not the USB host clock)
 - This synchronization may be needed across multiple devices.
 - Software only solutions to this problem are difficult
- Mechanism is provided for a device to control timestamp clock rate
 - For example - supports audio applications needing synchronization between multiple devices

Bus Interval Adjustment Mechanism (2)



- Device Notification to ask for small adjustments to the bus interval
 - Request unit is 4.069ps every bus interval
 - Minimum Adjustment is 8 HS bit times (16ns) over 1 second or $\sim(1/60\text{ppm})$
 - Request is for increment/decrement of N units, relative to the current bus interval
- Host must count clock ticks and add or remove 8 HS bit times from a bus interval at the appropriate count
 - Does not require changes to the underlying clock

Bus Interval Adjustment Mechanism (3)



- Device must not send additional bus interval adjustment request until it has waited long enough to observe the effect of the previous bus interval adjustment request on the timestamp value in subsequent ITPs
- Device must not make a single request for more than ± 4096 units
- Host honors requests from first device. Once request is received, Bus Interval Adjustment Control field in subsequent ITPs is set to the address of the requesting device
 - Software can instruct the host to follow requests from a different device.

Bus Interval Adjustment Mechanism (4)



- One unit bus interval adjustment (BusIntervalAdjustmentGranularity) requires the host to adjust bus interval timer by one 60MHz clock period (8 HS bit times) every 4096 bus intervals
 - The host can make the adjustment in a single bus interval with the rest of 4095 bus interval unadjusted. The bus interval adjustment is averaged over long period of time.
- Host must make adjustments in evenly distributed intervals. Following rules must be obeyed:
 - Difference between the number of eight HS bit time adjustments made in any bus interval shall not be greater than one
 - The distance in bus intervals between consecutive maximum adjustment bus intervals shall not be varied by more than one bus interval

Isochronous PING Scenario



- Software may enable appropriate link states (U1 or U2) based on service interval and throughput.
- Device may go into appropriate lower power link state when idle
 - For example - after all Isochronous transactions are completed in the service interval.
- Host sends PING to Isochronous device in advance of first transaction in the service interval. This transitions the path between host and device back to U0. Host assumes worst case exit latency in calculating launch time of PING.
- Intermediate hub (if any) may defer the PING, as the downstream link is in U1/U2. Hub forwards PING once the Link is in U0.
- Device responds with PING_RESPONSE when PING is received, after U0 is returned.
- Host schedules Isochronous transactions and transmits to device. Device is required to stay in U0 after PING_RESPONSE is transmitted, until tPingTimeout time ($2 * \text{Service Interval}$) or until another packet is received.

Host's Considerations in PING Transmission



- Latencies presented in hubs along path
 - U1 Exit Latency (bU1ExitLatency) if U1 is enabled and U2 is disabled
 - U2 Exit Latency (bU2ExitLatency) if U2 is enabled
 - Hub Header decoding delay (bHubHdrDecLat)
 - wHubDelay – Hub forwarding delay
- Latencies presented in targeted device
 - U1 Exit Latency (bU1ExitLatency) if U1 is enabled and U2 is disabled
 - U2 Exit Latency (bU2ExitLatency) if U2 is enabled
 - tPingResponse – from Device reception of a PING to initiating a PING_RESPONSE
- Latencies presented in Host
 - U1 Exit Latency (bU1ExitLatency) if link is in U1
 - U2 Exit Latency (bU2ExitLatency) if link is in U2
 - Isochronous Transaction Scheduling time – from PING_RESPONSE received to the time Isochronous transaction may be transmitted



Summary

- USB 3.0 Isochronous Protocol Preserves Key Characteristics of USB 2.0 Isochronous Protocol
 - No Retries. Simple Protocol
- Timestamps No Longer Broadcast (SOFs) to Improve Power Efficiency
- PING Mechanism To Allow Low Power Link States For Isochronous Devices
- Software Backwards Compatibility
- Maximum number of data transferred in one service interval is increased to 48K bytes.
- Mechanism for Device to Synchronize to External Clock