Interlaken Retransmit Extension Protocol Definition

---

28 June 2012

Revision 1.2

Terms and Conditions

Any company or individual wishing to use all or a part of this document may do so only if they relinquish their proprietary rights to information contained or referenced herein. You agree not to enforce those intellectual property rights against any other party implementing or advocating the implementation of this document.

YOU ACKNOWLEDGE THAT THIS DOCUMENT IS PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND. FOR THE SAKE OF CLARITY, ALL PARTIES USING OR ADVOCATING THE USE OF THIS DOCUMENT DISCLAIM ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, INCLUDING WITHOUT LIMITATION ANY AND ALL WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT.

IN NO EVENT WILL ANY PARTY THAT UTILIZES OR ADVOCATES THE USE OF THIS DOCUMENT BE LIABLE TO ANY OTHER PARTY FOR ANY INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THIS DOCUMENT, AND IN NO EVENT WILL THE CUMULATIVE LIABILITY OF ANY SINGLE PARTY FOR ANY AND ALL CLAIMS IN CONNECTION WITH THIS DOCUMENT EXCEED, IN THE AGGREGATE, ONE HUNDRED DOLLARS ($100.00). YOU AGREE THAT THIS PARAGRAPH IS AN ESSENTIAL BASIS OF THE USE OF THIS DOCUMENT.

Contents

Revision History

Revision 1.2 **20 April 2012**
• Section 2.1: Amended document to provision greater than 256 channels while maintaining retransmission capability
Revision 1.1 **26 September 2011**
• Section 2.1: Added example • Section 2.2.1: Improved clarity • Section 2.4: Added clarifications • Section 2.5: New section "Retransmit Timing Budget"
Revision 1.0 **8 December 2010**
• Initial public release of the document.

1 Introduction

Interlaken connects components via multiple high-speed serialized links. Data bits signals transferred across such serialized links are sometimes degraded enough that they are interpreted incorrectly at the receiver, which is called a “bit error”. The frequency of bit errors typically increases with increasing bit rates and with the number of physical lanes used in an Interlaken connection.

The Interlaken Protocol detects bit errors through its use of checksums. This optional retransmit extension defines a method to quickly resend the corrupted data and thereby minimize the system impact of link bit errors. Both sides of an Interlaken link must be retransmit capable for the link to operate in retransmit mode.

This extension does not alter the operation of the underlying regular Interlaken protocol, but makes use of the multi-use field in the burst control word and out of band flow control to manage the retransmit process.

2 Retransmit Protocol Extension

The extension provides a retransmission mechanism to allow hardware correction of bit errors on the Interlaken interface. It operates on top of the regular Interlaken protocol. Hence this protocol is defined as a super-set of the Interlaken protocol that operates on top of the standard, and does not significantly alter the underlying formats of the Interlaken protocol. The retransmission signaling is performed through redefining the use of the burst control word's multi-use field and parts of the out-of-band flow control. The protocol extension can be used on all Interlaken interfaces, regardless of lane widths and transceiver rates.

2.1 Overview of Protocol Extension

The protocol extension is broken into the transmitter and receiver pieces working together. Burst sequence numbers are applied to each data burst in the transmitter and carried across the Interlaken interface in the burst control word.

The receiver quickly detects a data burst error and signals to the transmitter to resend the corrupted data. The standard Interlaken CRC24 protection is used to detect burst errors. When the receiver detects an error, it signals to the transmitter to resend the data. The receiver is then responsible for using the burst sequence numbers to reconstruct the failing data.

The extension contains two redefinitions of the Interlaken protocol. They are:

1. **The Multi-Use field in the Burst Control Word is redefined as a burst sequence number**

The multi-use extension field in the burst control word is redefined as a burst sequence number. A burst sequence number is added by the transmitter into the multi-use extension field of every non-idle burst control word. The burst sequence number is defined as an 8-bit value that increments each n bursts (where the increment factor 'n' = 1, 2, 4, 8, 16, 32, default is 1). The receiver uses the burst sequence number to reconstruct the flow of data after an error has occurred which resulted in a retransmission.

For retransmission the multi-use field is utilized for the burst sequence number, it is therefore no longer available as a channel extension field for applications requiring greater than 256 channels (recall the channel number field is limited to a single byte). However, it is possible for larger channel count applications to still implement retransmission by reducing the number of bits utilized for the burst sequence number. In these cases the burst sequence increment factor 'n' must be evaluated to accommodate the round-trip delay of the OOB flow control plus the data pipeline to avoid sequence number aliasing. The Multi-use field is now shared with the most significant bits leveraged to extend the least significant bits of the adjacent channel field. The remaining bits of the multi-use field are retained as the retransmission burst sequence number.

Channel Number = {channel field, multi-use field[7:8-x]}

Sequence Number = {multi-use field[7-x:0]}

(where 'x' = 1, 2, 3)

There is a practical limit to the number of bits that can be repurposed from the sequence number to extend the channel field. As a rule of thumb the increment factor must be increased by a power of two for each additional channel number bit to avoid sequence number aliasing. Additionally the valid values for the increment factor are now extended to include the values 16 and 32 ('n' = 1, 2, 4, 8, 16, 32). With the practical limit of three bits available for extending the channel field, the current max channel count for retransmission applications increases from 256 channels to 2048 channels.

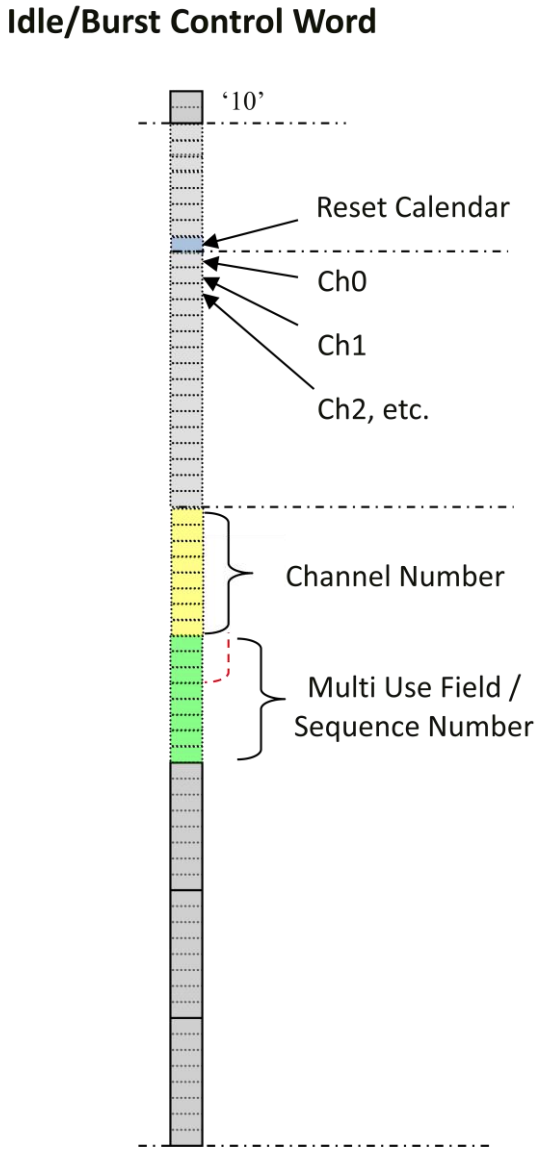

Figure 1: Channel Extension with Retransmission

2. One or more calendar entries in the OOB Flow Control interface are reserved for retransmission

The out-of-band flow control interface is optional in the Interlaken protocol, but for this extension it is mandatory. One or more calendar entry is reserved to signal a retransmission request, RetransReq, to the transmitter.

Only packet data burst control words contain a sequence number, idle control words and MetaFrame words do not (the multi-use field is set to all 0's for the idle control word). As the transmitter sends a data burst on the Interlaken interface, it also stores a copy of the burst with the burst sequence number attached in memory to be resent in case of an error. This memory is called the retransmit buffer. The amount of data the transmitter is required to hold in the retransmit buffer must meet the following 2 conditions:

- Must be less than 256 bursts of data to avoid sequence number aliasing (with n =1, 512 for n=2 etc.)

- Must contain enough data than can be sent to accommodate the round-trip delay of OOB flow control plus the data pipeline.

The first requirement is to guarantee there is not a wrap problem for the receiver following an error. The burst sequence number used by the receiver to reconstruct the data following an error is only 8 bits wide therefore there are only 256 possible values available (when n=1).

The second requirement is to guarantee the transmitter is still holding onto the errorred burst by the time the receiver has detected the error, signaled for its retransmission and finally received by the transmitter.

The receiver constantly checks for burst CRC24 errors and saves the sequence number of each burst that is received. When the receiver detects a burst CRC24 error, it latches the saved sequence number of the previous burst. The saved sequence number becomes the "last good" sequence number, in that it was the last error-free burst received. The receiver then signals for a retransmission by asserting the RetransReq signal on the OOB flow control interface. Note that a CRC24 error caused by an idle control word will cause a retransmission request.

When the transmitter detects the RetransReq asserted on the OOB FC interface, it stops sending new bursts of data but instead switches to sending the oldest saved burst of data it has in its retransmit buffer. The transmitter is not required to do any special handling of the retransmit data, only to guarantee that the order of bursts is the same and that the burst sequence numbers are consistent with the bursts as they were sent originally. It must send the entire contents of the retransmit buffer before starting to send new data. Proper formatting of Interlaken bursts must be maintained during the switch between the standard data bursts and the retransmitted data.

After the receiver has signaled for a retransmission, it should ignore any new burst CRC24 error for a specified amount of time. This amount of time is to allow any possible short term secondary errors to fall out and must be less than the best case delay for the OOB FC interface plus the data delay through the transmitter pipeline. During this period, the receiver continues to drop all incoming data bursts. The receiver must always continue to process all other framing errors except the burst CRC24 errors.

After the short delay of ignoring the incoming data stream, the receiver must start checking for new errors. At this point, it also begins checking for a match of the "last good" sequence number. The saved last good sequence number is checked against the sequence number on each burst received. All data bursts up to and including the burst that matches the last good sequence number are dropped. The bursts following the last good sequence number match are new data and should be forwarded downstream. One error case that must be handled is if the transmitter never sees the retransmission request (due to an error on the OOB bus). The receiver can guard against this by insuring that there is a discontinuity in the received sequence space where a discontinuity is an unexpected change in sequence number without CRC24 errors. For example, if the multiplier is greater than x1, and if a Burst Control Word with sequence number 14 is received followed by a Burst Control Word with sequence number other than 14 or 15 without any CRC24 errors in the Burst Control Words or in any of the Idle Control Words that may separate them, an unmistakable discontinuity is detected and a sure indication of the start of retransmitted data.

To the downstream logic, it appears as if the error never happened, except for an inactive period.

The retransmission feature only has a performance impact when an error is detected on the Interface. Under error-free operation there is no performance impact of this extension.

The specific implementation must deal with the possibility of an error while in the middle of a retransmission. A receiver should have a limit on how many retransmission requests it will ask for during a given retransmission event, once the limit is reached then a timeout would occur and the receiver would close all open packets and look for new SOPs.

Note that when an interface comes up, for the first time or after an error occurs which is severe enough to take the interface down, then the receiver accepts the first sequence number it receives from the transmitter as the correct sequence number for operation.

2.2 Retransmit Indication

At least one calendar entry in the OOB FC interface is reserved for retransmit request, RetransReq. The RetransReq signal indicates when an error was detected and a retransmission is requested.

There is no requirement for the exact location of the RetransReq in the OOB FC calendar, but both ends must agree on the location of the signal through the calendar mechanism.

The RetransReq signal is defined as:

A signal that is pulsed to generate a retransmission request from the receiver to the transmitter when an error is detected. Under error free operation the signal is a XOFF value (logic 0). When an error is detected the signal is asserted to XON (logic 1). The specific requirements of RetransReq are detailed in the next section.

The out-of-band flow control interface was chosen for sending the retransmission signal for the following reasons:

- The out-of-band flow control interface typically has a very low error rate compared to the data interface. The extension is used to address the error rate of the data plane, therefore sending the control over a similar interface is counter intuitive.

- Support simplex interfaces. In-band flow control requires two transceiver interfaces between the chips, by using out-of-band flow control unidirectional interfaces are supported.

2.2.1 RetransmitReq Timing Requirements

The RetransReq is at least one bit in the OOB flow control calendar and is an edge triggered signal. A retransmit request is valid on the assertion, set to Xon, of the RetransReq bit in the calendar. The bit in the calendar must be deasserted, set to Xoff, before another retransmit request can be signaled. If a RetransReq bit in a calendar is set to Xon for multiple RetransReq bits in a row, only the first occurrence triggers a retransmission.

Therefore, the length of the calendar can affect how quickly the transmitter can assert a 2nd retransmit request if needed. The minimum response time of issuing two retransmit requests is 3 calendar periods, with calendar 1 being the initial assertion of RetransReq, calendar 2 the deassertion and lastly calendar 3 being the second assertion (assuming that a single bit is used for retransmission requests, if there are multiple RetransReq bits per calendar period then the time scales down appropriately).

This could cause the blocking of a second request if it is required. This error condition should be handled in the receiver by not issuing the 2nd request and instead drop all bursts up to and including the last good burst and all packets in flight at that time. The following figure shows one example of the OOB flow control bus with a retransmission request bit.

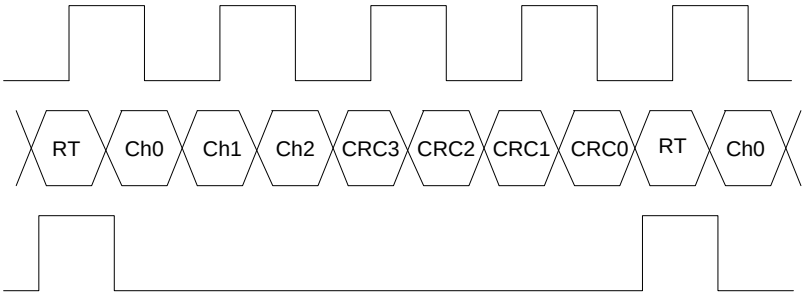

Figure 2: Retransmission request example

2.2.2 Examples

The following diagram has an example of normal operation with no errors. Note that once a burst is transmitted, it is also placed in the retransmission buffer, with the exception that idle control words are not stored in the retransmit buffer[1]. Also note that the data sent to the receiver comes from the tx buffer and not the retransmit buffer in this example.

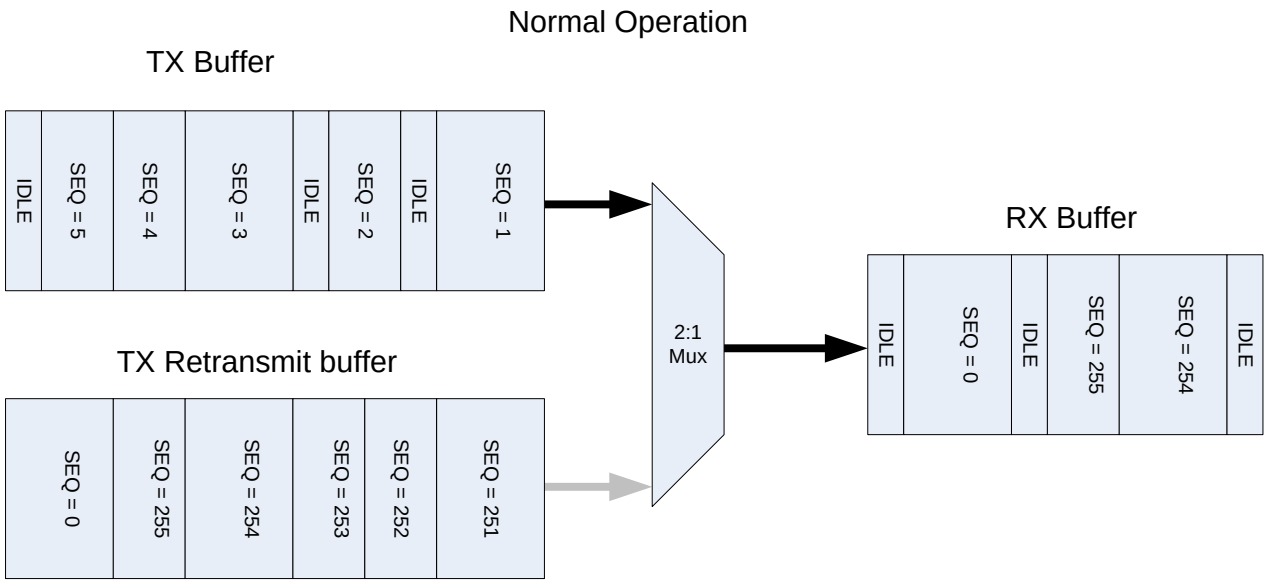

Figure 3: Normal Interlaken Operation

The following diagram has an example of an error being detected. In this case the receiver expected to see the sequence number 255 next; instead it saw a corrupted burst, so it immediately requests a

---

[1] If Idles were stored in the retransmission buffer, you can end up with the case where the last known (to the receiver) good sequence number has been overwritten from the retransmit buffer and therefore the receiver will not operate correctly. This would happen when there is little traffic being sent.

retransmission. The receiver will proceed to drop all bursts until it sees sequence #254 again, and then the receiver will resume accepting data.

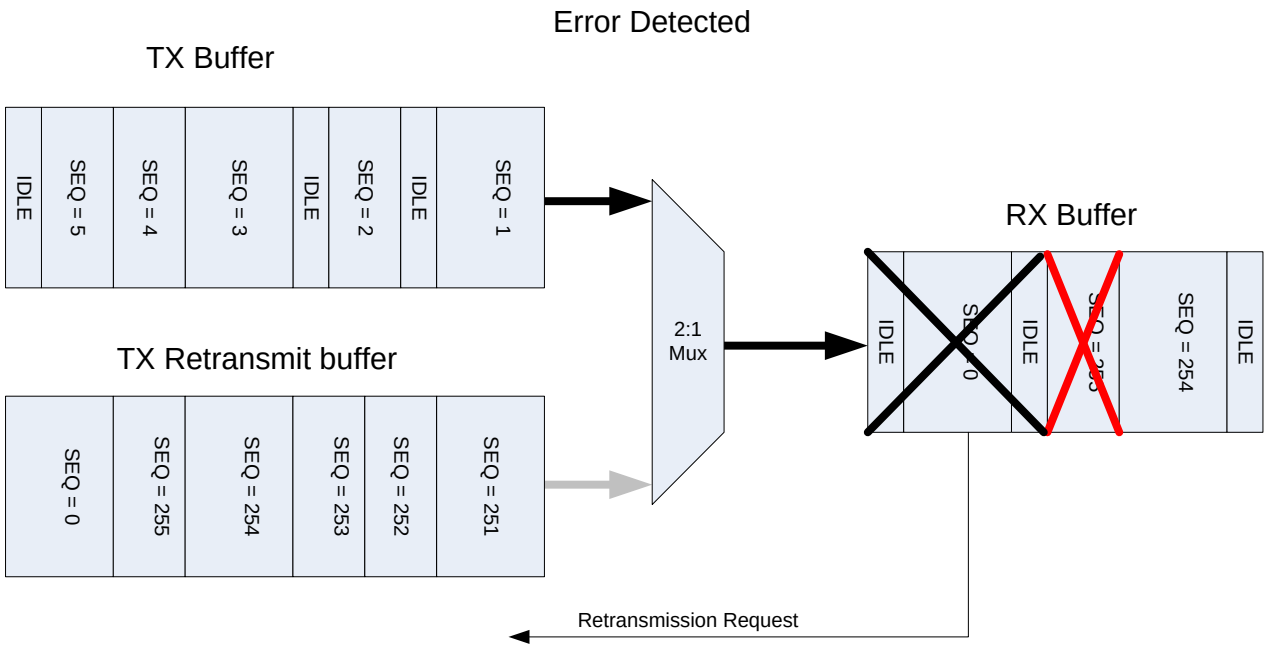

Figure 4: Interlaken Error Detected

The following diagram has an example of a retransmission in progress. In this case the receiver requested a retransmission after sequence #254. The receiver drops all bursts until it sees sequence #254 again, and then the receiver will resume accepting data, in this case it accepts the burst with sequence #255 (note that all data in the RX buffer in this diagram came from the retransmission buffer).

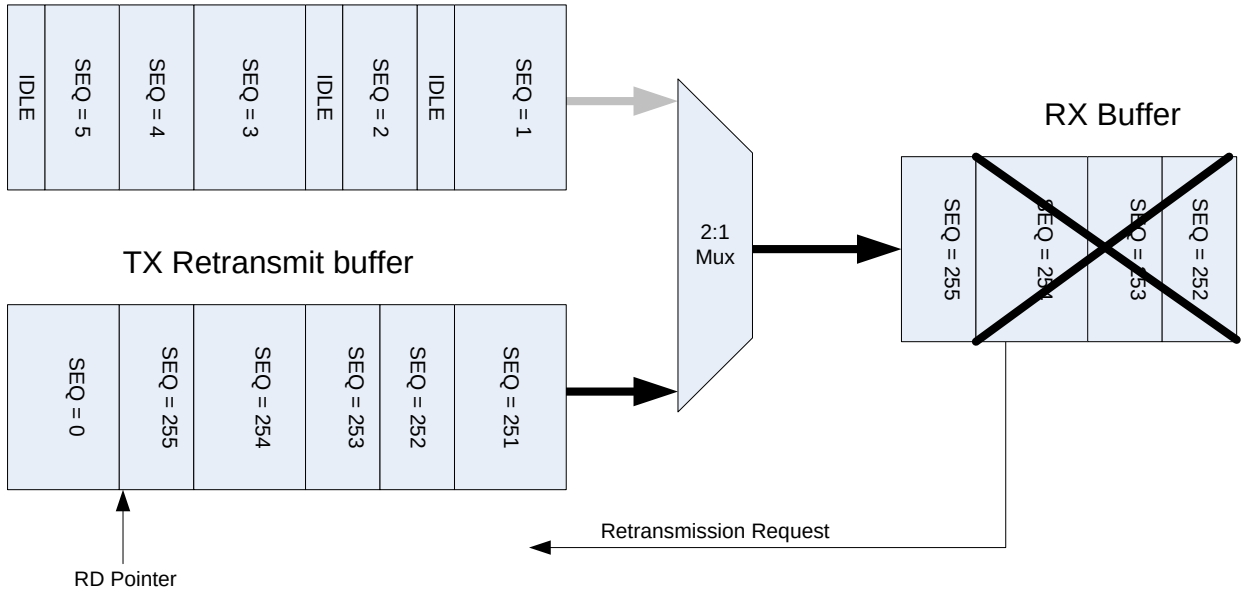

Figure 5: Interlaken Retransmission in Progress

The following diagram has an example of a normal operation but with n=2, where the sequence space is incremented by 2 (all other examples had n=1).

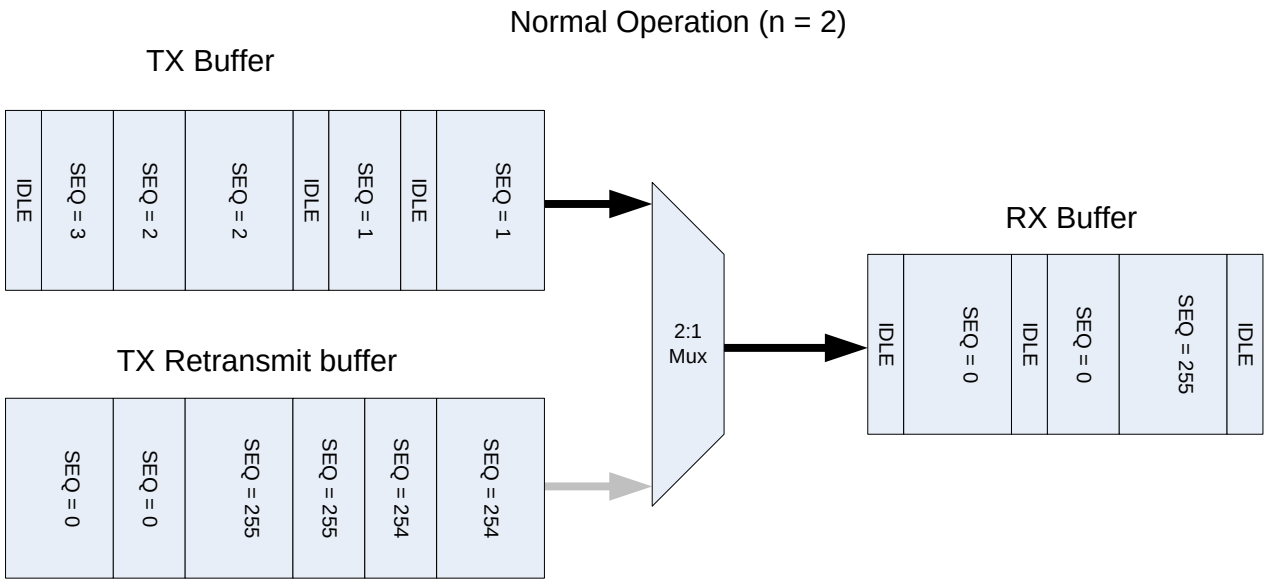

Figure 6: Normal Interlaken Operation with N=2

2.3 Extension Restrictions

The Interlaken retransmission extension protocol is a super-set of the Interlaken protocol but it does pose some limitations to Interlaken. The following is a list of restrictions for the Interlaken Retransmission protocol:

1. Out of band flow control must be used to signal for retransmission.

Out of band flow control is an option provided under the Interlaken spec. The Interlaken retransmission extension makes it a requirement with running with retransmission. The specific requirements of the additional flow control channels were described above in section 2.2.

2. The multi-use field is used to provide sequence numbers to track the bursts.

This means that the multi-use field cannot be used for any other function.

3. The receiver must be able to detect data burst errors and signal a retransmission request on the out of band flow control bus on a specific channel within a short period of time.

4. The transmitter must contain a retransmission memory in order to hold transmitted data.

- The transmitter is only allowed to send a maximum of 255 bursts for a single retransmit request (with n=1 for the count size, for n>1 multiply this by n).

- The transmitter must resend at least enough data to compensate for the worst case round trip delay for the amount of data in the pipeline plus the time for the RetransReq to be returned on the OOB FC interface.

5. The transmitter must follow all the Interlaken burst protocol requirements when initiating a retransmission and when finishing a retransmission and switching back to normal transmission.

- This means the BurstMin, BurstShort and BurstMax must be maintained

6. The transmitter is allowed to violate standard packet level protocol when initiating a retransmission.

- This means it is allowed to send a SOP to an already open channel or any data to a previously closed packet. The receiver must be able to handle these cases.

7. The transmitter must be able to accept a retransmission request on the out of band flow control bus on a specific channel and respond to it within a short period of time.

The specific channel used for the retransmit request must be programmable.

2.4 Recommendations

The following recommendations are intended to help ensure robust system design:

1. It is recommended that the receiver be capable of retrying a retransmission request for a given retransmission event. This guards against the two possible errors: 1) the transmitter never seeing the retransmission request (due to an error on the OOB bus), and 2) an unmistakable discontinuity was not detected in the received sequence stream.
Note that it is not required to first qualify a re-transmission request with the CRC4 since that would add to the latency of handling an error. Typically the OOB bus will be error free unless there is a catastrophic error such as a pin solder failure.
This is especially true if you have multiple re-transmission requests within a single calendar period. In general a misinterpretation of the retransmission request does not cause packet corruption or loss.

2. It is recommended that there be some minimum delay between retransmission requests to ensure the transmitter has sent a discontinuity. This becomes important when using a multiplier of x8 and bursts of BurstMax.

3. It is recommended that if the transmitter receives a retransmission request while still sending retransmitted data, the transmitter should as quickly as possible begin again to send the oldest saved data. This reduces the chance of a subsequent retransmission request for the same retransmission event.

4. It is recommended that the transmitter's retransmit buffer be able to hold enough data so that if a retransmission request is retried, the transmitter will still be holding onto the errored burst.

5. It is recommended that retransmit does not obey channel or link level flow control because any bursts in the retransmit buffer were already committed to be transmitted, so by definition, the RX must have been capable of receiving them. This would be true for both link level and per channel flow control.

6. When retransmitting data, and if in band flow control is used in an application, the flow control status sent in the retransmitted words should always have the latest flow control status (stale status should not be sent).

2.5 Retransmit Timing Budget

In order to ensure that a device designed by Vendor A will interoperate successfully with a device from Vendor B, the following retransmission timing budget is proposed as a guideline for designers. The following figure shows the breakdown of the components of the timing budget.

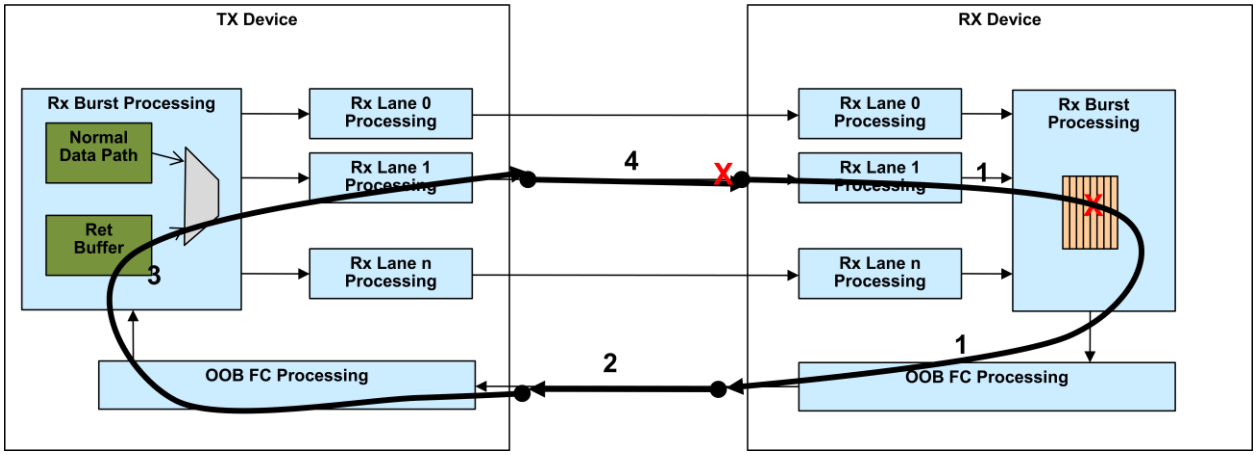

Figure 7: Retransmission Timing Example

As shown in the figure, the retransmission response time can be broken down into four segments:

1. Time from error appearing on the input of the receive device, to the retransmission request appearing on the OOB FC Interface at the output of the rx device (40% of the budget). This includes any delay due to the flow control calendar.

2. Transmission delay on the OOB FC Interface

3. The time from the notification of retransmission request to the last bit of the old data appearing on the interface (50% of the budget). Last bit is chosen since it stops incrementing the sequence space; implementations can have a delay from when retransmission starts and not impact the sequence space

4. Transmission delay of the data interface

Segment 1 of the response time is allocated 40% of the timing budget; segment 3 is allocated 50% of the timing budget. Segments 2 and 4 along with some level of safety margin are assigned the remaining 10% of the timing budget.

Example budget by assuming some parameters:

BurstShort = 64B, sequence space multiplier = 1

The total budget is then 64B * 1 * 255 = 16320 Bytes

Rx (1) has 40% of this, or 16320*.4 = 6528 Bytes

Tx (3) has 50% of this, or 16320 * .5 = 8160 Bytes

Bytes can easily be converted into time if the rate of the interface is known.

2.6 Test Capability

It is suggested that an implementation of the retransmission protocol support the following test capabilities:

- The capability to insert a bit error under software control.
- The capability to force a retransmit request under software control.