Date: July 28, 2003

To: T10 Committee (SCSI)

From: George Penokie (IBM/Tivoli)

Subject: End-to-End Error Cases

1 Overview

As part of the discussion on end-to-end data protection the question of what errors are being detected has come up several times. This document describes the error cases which end-to-end data protection protects against and the mechanism for detecting those errors.

2 Mishandling of blocks within a Device

This is a case where part of the data is lost due to improper handling within a device that temporarily stores that data. An example would be a device that improperly caches received data before retransmitting. An example is shown in figure 1.

This type of a device would look like a target to the device sending the data and would look like an initiator when transmitting the data. There is currently no method defined in SCSI that would allow this type of lost data to be detected. However, as shown in the example, placing an incrementing tag value with each block could detect this class of error.

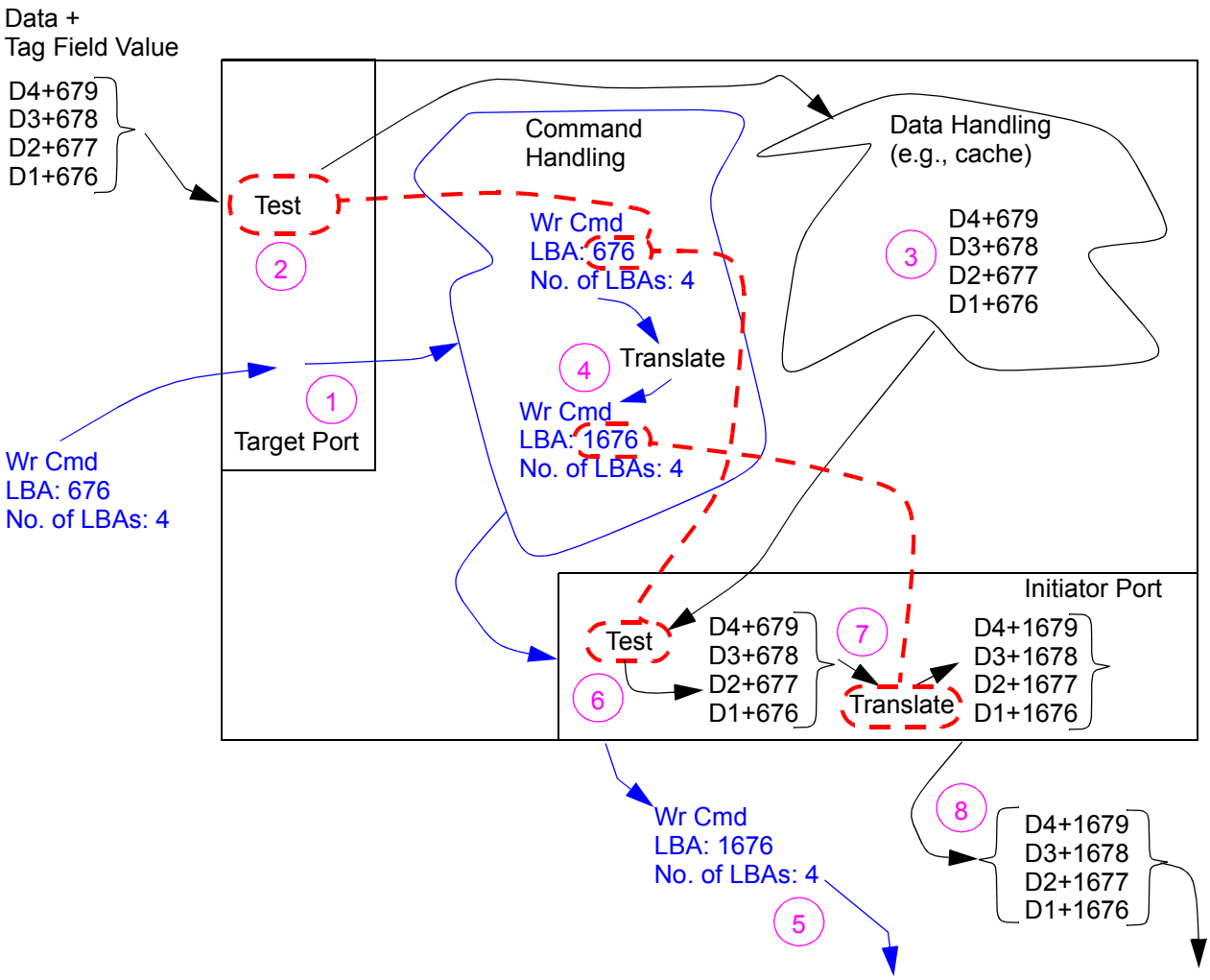


Figure 1 — Write command cache data loss example

The steps in figure 1 are:

1) Receive and save write command;
2) Receive data and test tag against information in write command (see figure 2);
3) Store data in temporary storage (e.g., cache);
4) Translate LBA to the destination LBA and create a new write command;
5) Transmit the new write command to the destination logical unit;
6) Move data from temporary storage and test tag (see figure 3);
7) Translate tag to match LBA in new write (for types of translate errors see 2.2); and
8) Transmit data and tag to destination.

Table 1 — Cache error detection

Error [a]	Result of Error	Detection Method during Write	Detection Method during Read
Mishandling of data words in the cache.	A corrupted data block is retrieved from cache.	Detectable before transmission if: a)there is protection placed on each block at the source based on the contents of the block; b)the protection is not regenerated as the block moves from the original source to the final destination; and c)the protection is checked before transmitting or it is checked at a write destination.	Detectable at the read destination if: a)there the protection placed on each block at the original source (i.e., the source of the write) was based on the contents of the block; b)the protection is not regenerated as the block moves between the original source to the final destination; c)the protection was written, unchanged, to the media; and d)the protection is checked during the read.
Mishandling of data blocks in the cache.	The wrong data block is retrieved from cache.	This can be detected if: a)there is a field that is written on each block of data (e.g., LBA); b)the value in the field is not determined by the data in the block; c)the value in the block is known by the device (e.g., relates to LBA in the received CDB); d)is not changed until after the block is removed from cache; and e).is tested against a known value after being removed from cache and before being transmitted. or If the device translates and the next device (i.e., the target) to receive the block tests the translated value against a known value.	This can be detected by an initiator if: a)there is a field that was written on each block of data (e.g., LBA); b)the value in the field is not determined by the data in the block; c)the value in the block is known by the device (e.g., relates to LBA in the received CDB); and d).is tested against a known value after being received.
The wrong data block is received.	Bad data is read.	This can be detected at a target if: a)there is a field that is written on each block of data (e.g., LBA); b)the value in the field is not determined by the data in the block; c)the value in the block is known by the device (e.g., relates to LBA in the received CDB); and d).is tested against the known value after being received.	This can be detected at an initiator if: a)there is a field that was written on each block of data (e.g., LBA); b)the value in the field is not determined by the data in the block; c)the value in the block is known by the device (e.g., relates to LBA in the received CDB); and d).is tested against a known value after being received.

[a] An example of a error would be if a pointer into the cache was misdirected.

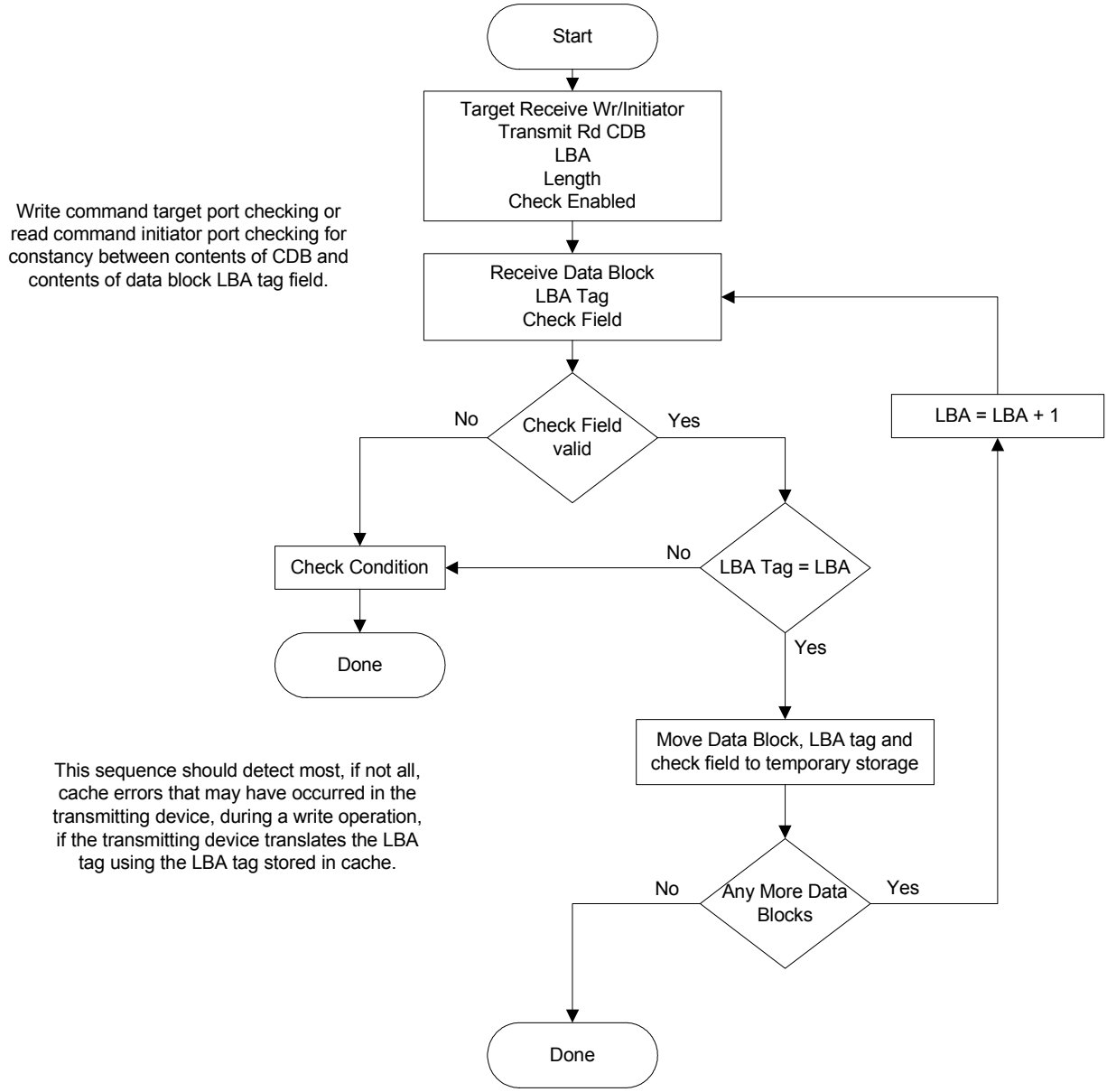


Figure 2 — Target port/initiator port receive data block checking flowchart

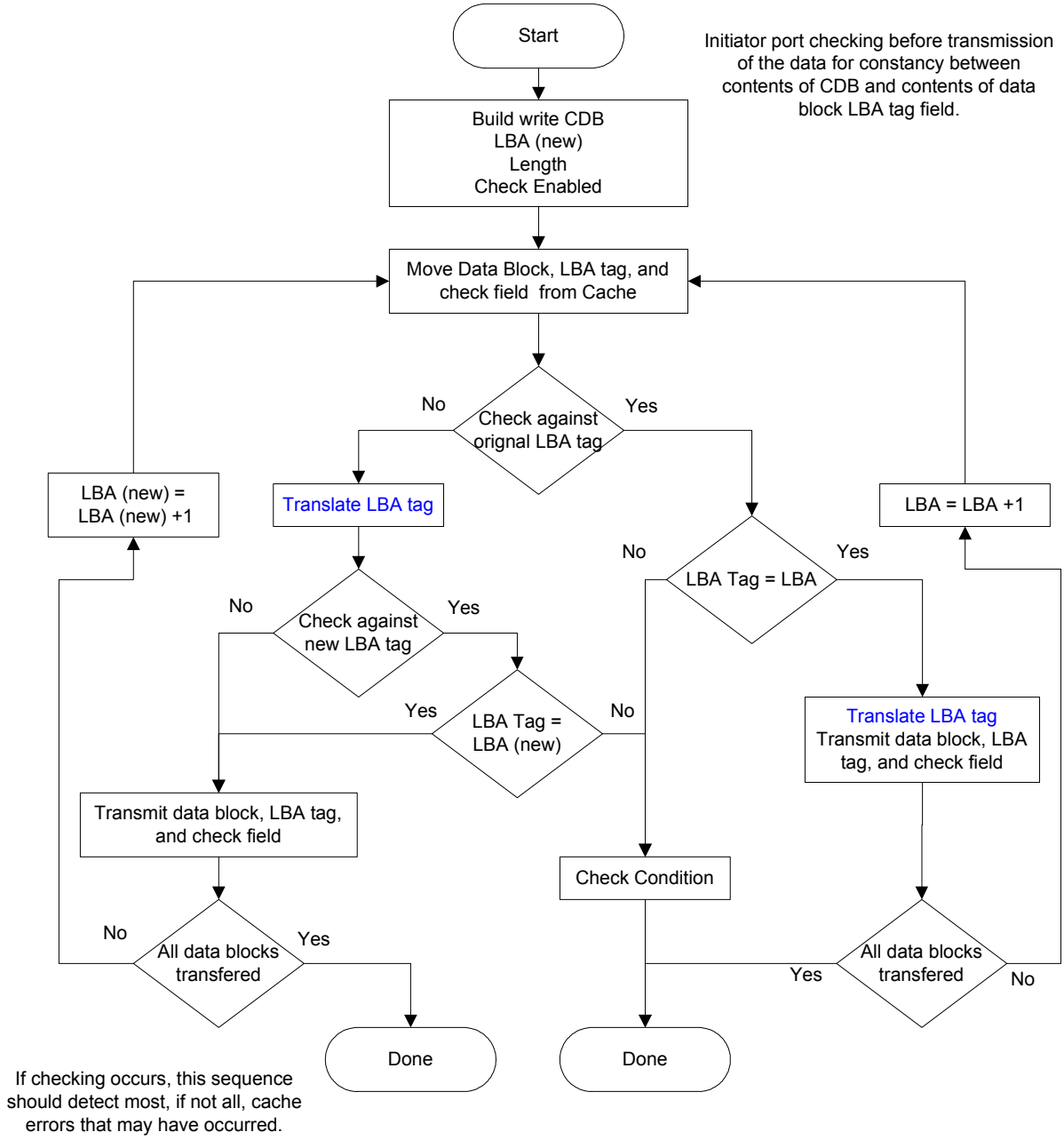


Figure 3 — Initiator port transmit data block checking flowchart

2.1 LBA tag translation vs. no LBA tag translation at initiator port

This clause looks into the question of whether the LBA tag information should be carried separately in the CDB or tied to information already contained within a CDB (e.g., all or part of the LBA). The flowchart shown in figure 3 assumes the LBA tag information is carried within a normal CDB. In that case the blue text indicates the action that would be taken by an initiator to accomplish the verification of information before transmission. If the initiator passes the LBA tag information in the CDB (i.e., not within the current CDB information) that the steps indicated by blue are not required or allowed.

Table 2 lists a comparison between LBA tag translations and LBA tag pass-through.

Table 2 — LBA tag translation vs. pass-through

Issue	LBA Tag translation at initiator	LBA Tag pass-through
Translate the LBA Tag in the logical block after removing block from cache and before transmitting	Required Requires that the initiator translate information associated with each data block before transmitted the data block	Not Allowed Advantage in that there is no requirement that the initiator do anything to the data block.
CBD size	No change to CDB size. LBA tag information is carried in the already defined LBA field	Requires larger CDBs to be defined for all commands that will use the LBA tag field
Error detection	Detects a caching error	Detects a caching error
Location of error detection	Either at the transmitter (i.e., initiator port) or at the receiver (i.e., target port) during a write or either at the transmitter (i.e., target port) or at the receiver (i.e., initiator port) during a read.	Either at the transmitter (i.e., initiator port) or at the receiver (i.e., target port) during a write or either at the transmitter (i.e., target port) or at the receiver (i.e., initiator port) during a read.
Reading device's knowledge of LBA tag	Always knows because the LBA being read is the LBA tag value	Has to have knowledge of the algorithm used by the writer of the data or do no checking on read
Coalescing of writes [a]	Permitted because the translated LBA tags would be contiguous.	Not permitted because the passed through LBA tags would no longer be sequential. [b]
Out of band initiated copy services	No problem because the LBA tag would be the same as the LBA being requested in the copy service	The copy service would not be able to test for errors unless it has knowledge of the algorithm used by the writer of the data or do no checking on read

[a] Coalescing of writes refers to when the strips from multiple stripes are combined and written to one device. This is usually done to minimize the number of write operations to a single device.

[b] The problem with passing through LBA tags in this environment is that the data blocks for a single write CDB would not contain a continuous sequence of numbers (i.e., the value in the LBA tag would jump several values at each strip boundary) making it difficult to test for errors.

2.2 Storage device LBA translation error

This is a case where a storage device (e.g., RAID 5 controller) that receives a write to an LBA that is translated into another LBA and then transmitted to another storage device (e.g., disk drive) and that translation fails. An example is shown in figure 4. Although this example is specific to a RAID controller it is valid for any target device the translates received logical unit numbers from the received value to a new target port/logical unit destination.

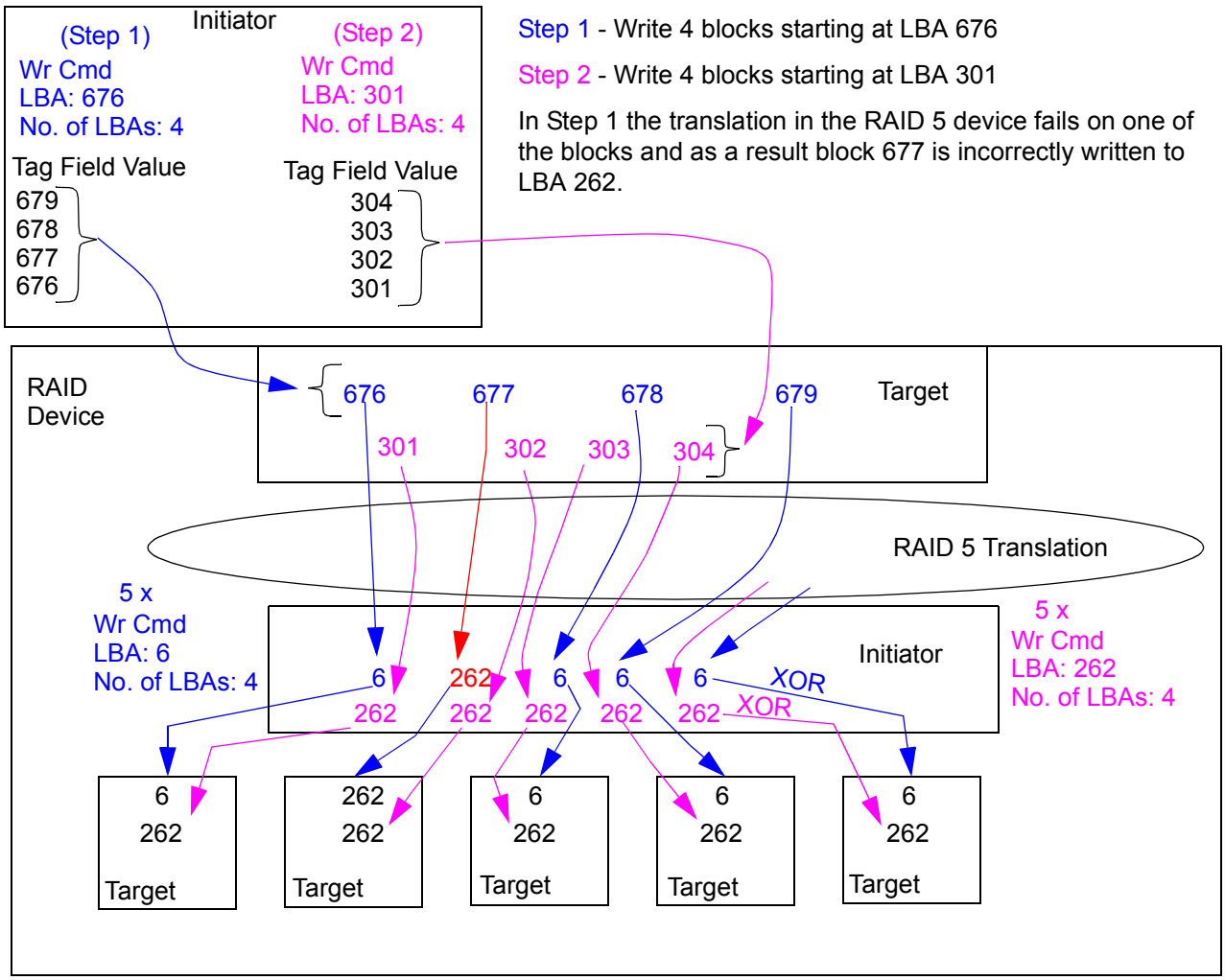


Figure 4 — Storage device LBA translation error example

Table 3 describes the methods for detecting the error case shown in figure 4 if only step 1 occurs.

Table 3 — LBA translation error detection (Step 1 only case)

Error	Result of Error	Detection Method during Write	Detection Method during Read
Fixed translation error (i.e., LBA x always translates to LBA z instead of LBA y)	All requests to read or write LBA x results in a read or write of LBA z	Not possible as everything looks good to the targets	Not possible as everything looks good to the initiators. This is not really an error as the data read is the data that was written.
Intermittent or random translation error (i.e., LBA x sometimes translates into something other that LBA y)	Requests to read or write LBA x results in a read or write some LBA possibly not LBA y (i.e., read or write of wrong data).	Not possible as everything looks good to the targets	This can be detected if: [a] a)there is a field that is written on each block of data (e.g., generation information); b)the value in the field is not determined by the data in the block; c)is not changed as the block moves from the original source to the final destination; and d).the device reading the block knows the algorithm used to write the field.
An intermittent write error with a read of the correct LBA	Request to read the LBA may result in old data being returned (i.e., read of stale data)	Not possible as everything looks good to the targets	This can be detected if: [a] a)there is a field that is written on each block of data (e.g., generation information); b)the value in the field is not determined by the data in the block; c)is not changed as the block moves from the original source to the final destination; and d).the device reading the block knows the algorithm used to write the field.

[a] A solution may be to define a vendor specific field of size n. Then assume that only applications that have to have knowledge of the fields contents should use it.

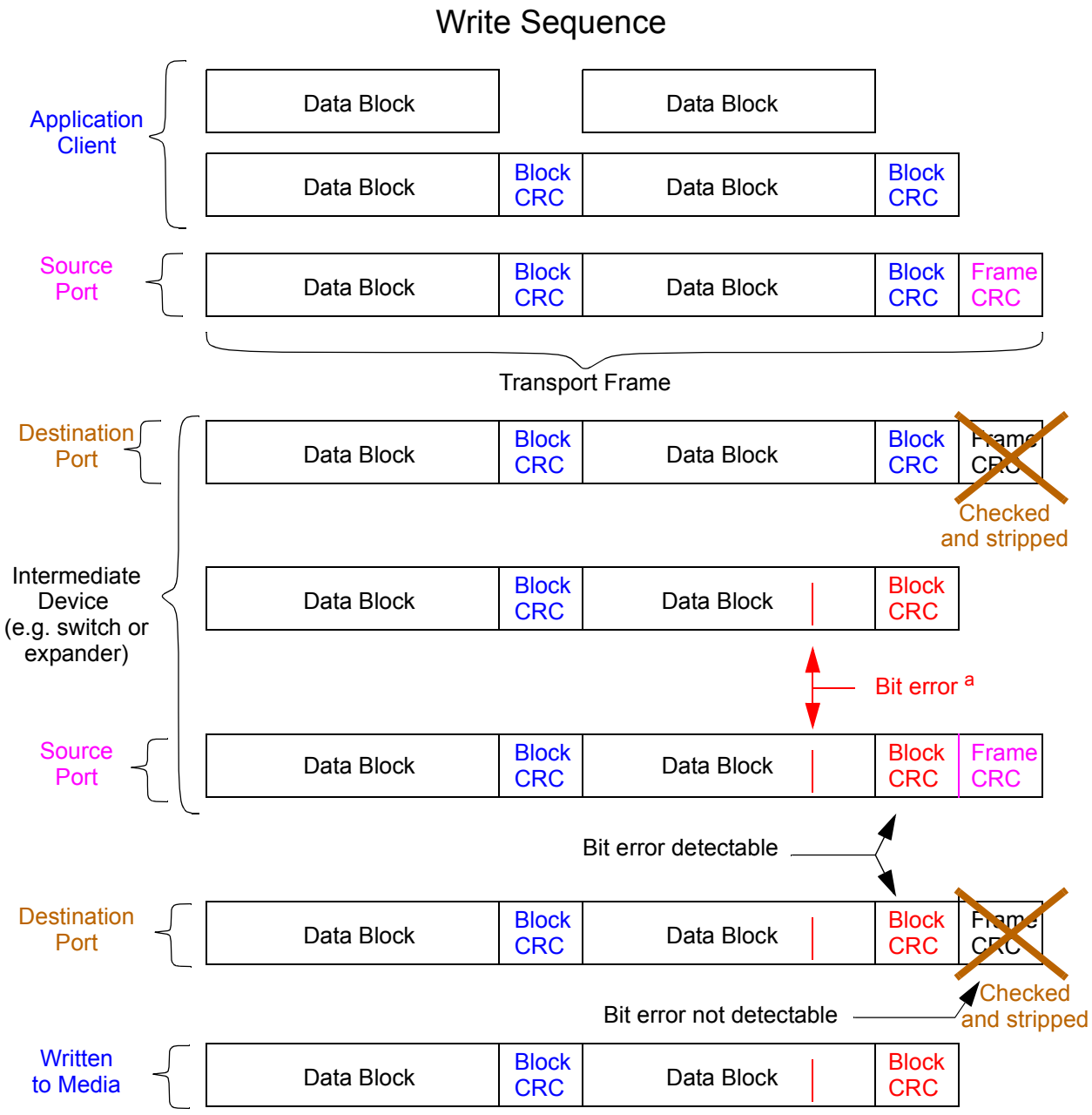
Table 4 describes the methods for detecting the error case shown in figure 4 if step 1 and step 2 occur.

Table 4 — LBA translation error detection (Step 1 and 2 case)

Step	Error	Result of Error	Detection Method during Write	Detection Method during Read
1	Fixed translation error.	All requests to read or write LBA x results in a read or write of LBA z	Not possible as everything looks good to the targets	This can be detected if: a)there is a field that is written on each block of data; b)the value in the field is not determined by the data in the block; c)is not changed as the block moves from the original source to the final destination; and d).the device reading the block knows the algorithm used to write the field.
2	No errors but the data is written to the same LBA as was incorrectly written to in step 1			

3 Bit Errors Inside a Device

This is a case where the data in a block is changed while it is being processed and/or stored in a device (e.g., a switch) that receives and then retransmits the block. The protection (e.g., CRC) that is used to protect the transmission of the block is generated at the transmitter and checked/stripped at the receiver. As a result there is no standardized protection on the block between receivers and transmitters. An example is shown in figure 5.



[a] A bit error may also occur within the CRC

Figure 5 — Device bit error example

Table 5 describes the methods for detecting the error case shown in figure 5.

Table 5 — Bit error detection

Error	Result of Error	Detection Method during Write	Detection Method during Read
A bit transposition occurs in a device in the data path that retransmits data	A block of data is corrupted.	Detectable at the write destination if: a)there is protection placed on each block at the source based on the contents of the block; b)the protection is not regenerated as the block moves from the original source to the final destination; and c)the protection is checked at a write destination.	Detectable at the read destination if: a)there the protection placed on each block at the original source (i.e., the source of the write) was based on the contents of the block; b)the protection is not regenerated as the block moves between the original source to the final destination; c)the protection was written, unchanged, to the media; and d)the protection is checked during the read.

4 Lost Frame

This is a case where a frame is lost during transmission. The frame level protection (e.g., offset) that is used to protect the transmission of the frame is generated at the original transmitter and checked at the final receiver. This should provide enough protection on the application client side as the offset value is set by the application client, if the transport protocol supports this feature. An example is shown in figure 6.

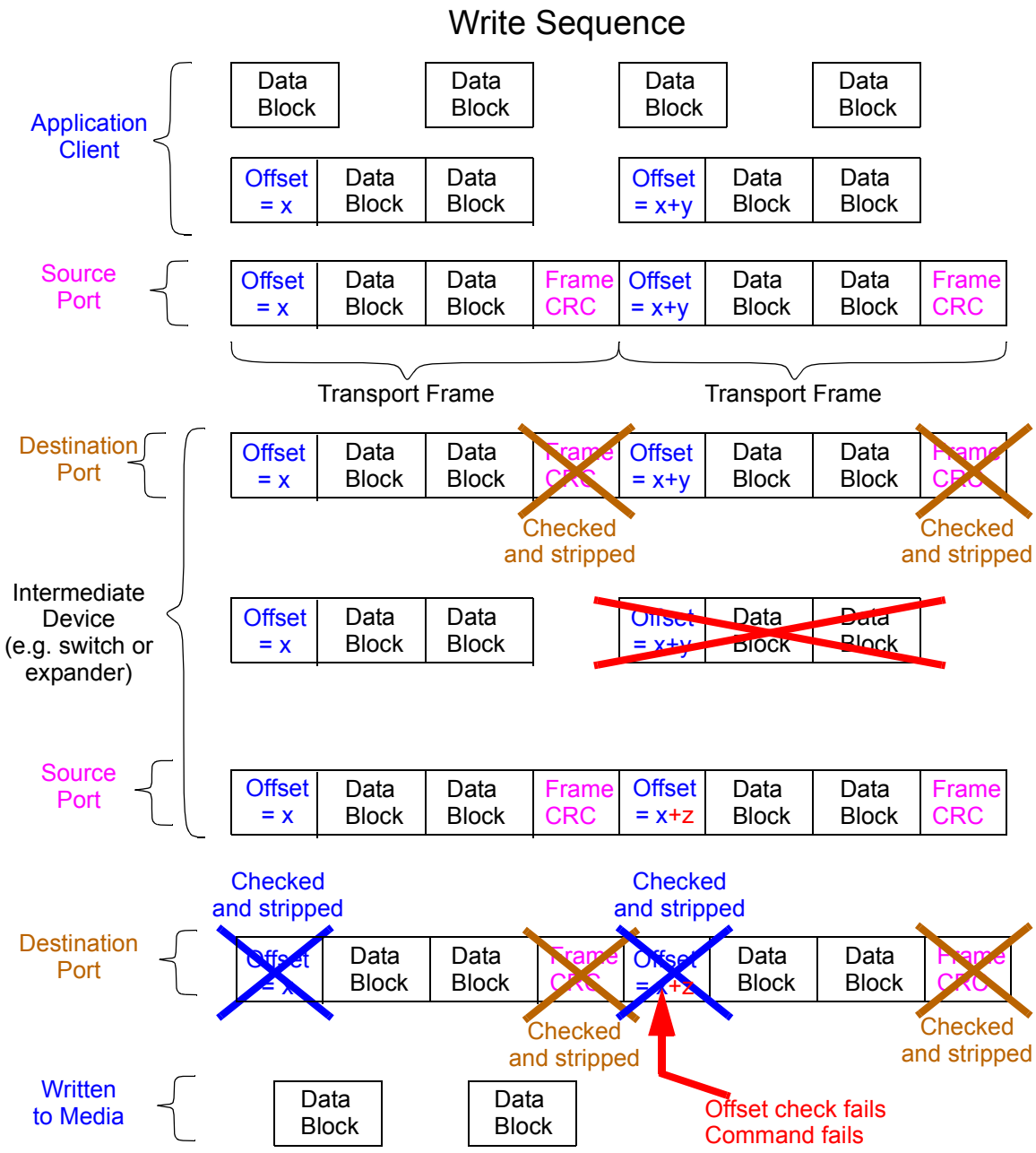


Figure 6 — Lost frame example

Table 6 describes the methods for detecting the error case shown in figure 6.

Table 6 — Lost frame detection

Error	Result of Error	Detection Method during Write	Detection Method during Read
A frame is lost in a device in the data path.	A frame of data is lost.	Detectable at the write destination if: a)there is protection placed on or in each frame at the source that starts at a known value and increments on every frame boundary; b)the protection is not regenerated as the frame moves from the original source to the final destination; and c)the protection is checked at a write destination.	Detectable at the read destination if: a)there is protection placed on or in each frame at the source that starts at a known value and increments on every frame boundary; b)the protection is not regenerated as the frame moves from the original source to the final destination; and c)the protection is checked at a read destination.