CRC position and CRC checking for Data Integrity

Pat Thaler
Agilent Technologies
1 October 2003

There are two proposals for Protected Block Format, T10/03-336r0 and T10/176r5, which differ primarily in the position of the CRC field within the block. Some have suggested that it is excessively difficult to check the CRC field when it is not contiguous to the covered data because the CRC calculation will have to pause while skipping over the non-covered fields.

This paper will show that it is easy to check a CRC that is not contiguous with the covered data. It will do this by demonstrating two methods for checking the CRC when the CRC is not contiguous with the protected data block. The methods do not require pausing the CRC calculation to skip over non-covered data. There are, of course, many other ways this could be implemented.

Before describing the methods, it is interesting to note that the SAS draft requires a CRC check that pauses its calculation for cycles when it doesn't have data. A SAS frame arrives with ALIGN and NOTIFY primitives interspersed with the data. The ALIGN and NOTIFY primitives are deleted and inserted by the physical layer so any clock cycle may bring a data dword or a primitive.

Method 1: Comparator

There are two ways to check a CRC of a message. One is to run the message concatenated with the CRC through the checker and compare the result to a constant. In the case of the data integrity proposal, the constant is zero. Checking the result in the CRC checker requires a 16-input NOR.

The other equally valid method is to run the message through the CRC checker and then compare the resulting value to the CRC value. This requires 16 two-input XOR gates plus a 16-input NOR.

Therefore, one implementation would run the data block through the CRC checker and compare the result to the CRC field value. In this case, the CRC check does not have to run over non-contiguous data.

Method 2: Using an already existing save and load operations

Note that SCSI data blocks for frame-based transports such as Fibre Channel may be split across frames. These frames may be interspersed with other frames from other tasks carrying data with data integrity. Therefore, if an implementation is going to calculate the CRC as the data arrives, it will need the ability save the partial result from the calculation when a frame ends in a partial block. Later when a frame arrives with the rest of the

block, the implementation will need to load the partial result into the CRC checker and resume calculation.

The same save and load operations that are used to deal with a block split across frames can be used to save the value in the CRC generator at the end of the data block and reload it at the beginning of the CRC field.

Summary

Two methods have been shown to check CRC without the need to pause the CRC calculation. Method 1 requires 16 XOR gates. Method 2 uses existing functionality. It is also noted that SAS assumes the ability to pause CRC calculation.

Therefore, implementation difficulty is not a factor that distinguishes the two proposed field orders.