DC-Balanced Character Encoding for SpaceWire

Cliff Kimmery

Introduction

- Many applications migrating to SpaceWire from alternate communications protocols (e.g. MIL-STD-1553) provide much greater tolerance for long-term and transient differences between ground references
  - Standard SpaceWire has limited support for applications requiring galvanic isolation between link endpoints
    - Limited common-mode tolerance of ANSI/TIA/EIA-644 LVDS devices
    - Unbalanced character-level encoding method established by ECSS-E-ST-50-12C, Clause 7 [1]
- Proposed solution: a practical alternative character-level encoding method
  - Support galvanic isolation using conventional Alternating Current (AC)-coupling circuits
  - Maintain current ANSI/TIA/EIA-644 LVDS technology
  - Maintain the clock recovery benefits of Data-Strobe encoding
  - Provide error detection comparable to the standard SpaceWire parity check
  - Minimize the impact to link bandwidth efficiency

[1] ECSS, “Space engineering - SpaceWire - Links, nodes, routers and networks”, ECSS-E-ST-50-12C, 31 July 2008, pages 52-56, http://spacewire.esa.int/content/Standard/ECSS-E50-12A.php

Introduction

- The result is a class of codes that simultaneously Direct Current (DC)-balance both the Data and Strobe bit streams
  - Members of the class with a larger code size increase encoding overhead
  - Members with a smaller code size have greater algorithm complexity
    - To track running disparity, etc.
  - Examples of the class
    - 10 bits – smaller code size and complex encoding method
    - 12 bits – large code size and simple encoding method
    - 14 bits – large code size and complex encoding method

Note that the term **character** is used in this presentation as defined by the SpaceWire standard and includes data characters and control characters

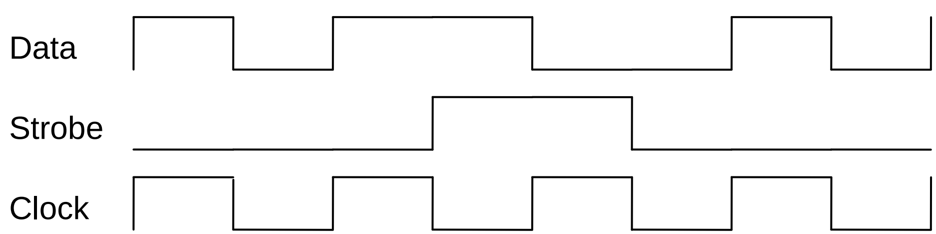
- The term **code** is defined as a binary value used to represent a character when transmitted on the SpaceWire link
- In standard SpaceWire, a character and the corresponding code are identical

Outline

- SpaceWire character encoding background
- DC-Balanced character encoding background
- Development of DC-Balanced character encoding
  - Evaluation of code value candidates
  - Code set size
  - Code set selection results
  - Effects of code length on bandwidth efficiency
  - Error detection capability
- Results and conclusions
- Summary

SpaceWire Character Encoding Background

- SpaceWire character-level encoding starts with Non-Return-to-Zero (NRZ) encoded ten-bit characters serialized as the Data signal
  - The Strobe signal is generated from the Data signal by Exclusive OR (XOR) with an alternating binary one-zero pattern of identical length (see the figure)
  - The alternating one-zero pattern represents a one-half-rate clock with transitions corresponding to the bit intervals of the Data signal
    - Commonly known as a Double-Data Rate (DDR) clock
- Because standard SpaceWire encoding uses raw binary values to form NRZ-encoded characters for the Data signal, the degree of DC balancing achieved is determined by the character sequence transmitted
  - Introducing a balanced Data signal bit stream doesn’t automatically create a balanced bit stream for the corresponding Strobe signal



DC-Balanced Character Encoding Background

- Each transmitted bit stream (Data and Strobe) must have the same transition-rich characteristics so that the frequency content is equivalent
  - The DC-Balanced Data character encoding must result in comparable transition density and run-length for Data and Strobe
- The initial transition density and run-length benchmarks were established from the well-known 8b10b encoding method
  - 8b10b encoding guarantees a transition-rich data stream so that the receiving device can recover clock from the serial data (embedded clock)
  - Definition of transition rich [2]

> For every 20 successive bits transferred
> 1. The disparity (the difference in the number of ones and the number of zeros) cannot be more than two
> 2. There cannot be more than five ones or five zeros in a row

- Why not use the 8b10b encoding method directly?
  - Because of the need to DC-balance both the Data and Strobe signals simultaneously
  - The Strobe bit stream generated from an 8b10b encoded Data bit stream does not have adequate DC-balancing characteristics

[2] Alex Goldhammer and John Ayer Jr., “Understanding Performance of PCI Express Systems”, September 4, 2008, page 2, http://www.xilinx.com/support/documentation/white_papers/wp350.pdf

Development of DC-Balanced Character Encoding

- Convert the 8b10b run-length benchmark criterion to a form that allows easy evaluation of candidate binary values
  - The number of consecutive same-value bits within any two successive encoded characters should be five or less
    - Including the boundary between the two characters
- Translating the disparity benchmark criterion is nontrivial when the code size is greater than 10 bits
  - Use a somewhat stronger approximation
    - The disparity of any single code must be one or less (corresponding to two consecutive codes with combined disparity of two or less)
- Each code that has nonzero disparity must have at least one alternate code with the opposite (negative) disparity
  - Any character can occur in combination with any other character (including itself)
  - A binary value with an even number of bits has even disparity 0, 2, 4, etc.
  - A binary value with an odd number of bits has odd disparity 1, 3, 5, etc.

Evaluation of Code Value Candidates

- Selection for membership in a candidate DC-Balanced code set was based on the following parameters
  1. Maximum disparity (8b10b criterion: ≤ 2 for any 20-bit sequence)
     - The disparity of each code in the code set must be no greater than the maximum disparity parameter
  2. Maximum run-length (8b10b criterion: ≤ 5 for any 20-bit sequence)
     - The number of consecutive one bits or zero bits must be no greater than the maximum run-length parameter
  3. Maximum leading run-length
     - The number of consecutive leading one bits or zero bits must be no greater than the maximum leading run-length parameter
  4. Maximum trailing run-length
     - The number of consecutive trailing one bits or zero bits must be no greater than the maximum trailing run-length parameter
- The code boundary run-length issue was addressed by including the maximum leading and maximum trailing run-length parameters

Maximum Disparity	Maximum Run-Length	Maximum Leading Run-Length	Maximum Trailing Run-Length
1	5	2	3

Code Set Size

- The SpaceWire character set has 256 data characters and 4 control characters
  - At least 260 distinct code pairs are needed to encode the complete character set
  - A set of code pairs with a nonzero maximum disparity characteristic must include a minimum of 520 distinct code pairs
    - Each pair must be matched with another pair with the opposite disparity to allow representation of the same SpaceWire character with either pair
  - Odd-length codes must always include a minimum of 1,040 distinct code pairs
    - Non-zero disparity is inherent
    - The nature of Data-Strobe encoding causes two identical odd-length Data codes in succession to produce different Strobe codes
    - It is convenient to require that successive identical SpaceWire characters be encoded to different odd-length codes

Code Set Selection Results

- Code Result by Length

Bits	Maximum Disparity	Maximum Run-Length	Set Size
Even length codes require a set size of 260 (disparity 0) or 520			
10	4	7	552
12	0	6	284
14	2	4	1144
16	0	4	260
Odd length codes require a set size of 1,040			
11	3	8	1048
13	1	7	1040
15	1	4	1188

Note: the Maximum Run-Length is the greater of the Run-Length and the sum of the Leading Run-Length and the Trailing Run-Length

Note: the Set Size is the number of codes that met the corresponding evaluation criteria. The codes to be used are chosen from the full set as desired.

Effects of Code Length on Bandwidth Efficiency

- Standard SpaceWire characters have differing lengths depending upon function
  - Any code length greater than the standard length can have a significant impact on link bandwidth efficiency
- Overall SpaceWire link efficiency is dynamically determined by the mix of SpaceWire characters transmitted
  - Data characters and FCT characters dominate link traffic
  - End-of-packet characters are relatively rare
  - Time code characters are rarer still
  - Null characters are only used to keep the link active

Error Detection Capability

- Standard SpaceWire adds a parity bit to each encoded character to detect transmission bit errors
  - The error response is to disconnect the link, report the error and attempt to reconnect the link (the same approach is used to recover from all link errors)
- The DC-Balanced codes have intrinsic characteristics that make transmission error detection straightforward
  - The error response defined by standard SpaceWire is unchanged
  - The DC-Balanced decoding mechanism has inherent error detection capability since an unrecognized code is considered an error
    - A transmission error occurring in a DC-Balanced code must convert that code to a different valid code for the error to be undetectable
    - The members of a DC-Balanced code set can be selected to have sufficient Hamming distance to prevent many transmission errors from being undetectable
  - Either the Data code or the Strobe code can be decoded to the equivalent SpaceWire character
    - In cases where an adequate Hamming distance is not achievable, the Data code and the Strobe code can be independently decoded and then compared to detect most transmission errors
  - The two mechanisms can clearly be combined to provide very robust transmission error detection

Results and Conclusions

- The 15-bit length code set is the most bandwidth efficient of the DC-Balanced code sets that fully met the 8b10b benchmark criteria
  - The bandwidth efficiency is at best 63% that of standard SpaceWire
- The other fully qualified code set (16-bits) is less efficient than the 15-bit code set, but is simpler to implement

- Relaxing the benchmark criteria allows use of DC-Balanced code sets with greater bandwidth efficiency
  - The effects of relaxing the benchmark criteria on link performance must be determined by signal integrity analysis and experimentation
- The 12-bit length DC-Balanced code set has the advantage of zero-disparity implementation simplicity
  - Exceeds the run-length benchmark criterion by 20%
  - The bandwidth efficiency can be improved to approximately 80% by choosing an FCT code with at most 6-bit length

Results and Conclusions

- The 10-bit DC-Balanced code has the best bandwidth efficiency
  - Improves the bandwidth efficiency to within 5% of standard SpaceWire when using a 6-bit FCT code length
  - Misses the disparity criterion significantly
    - The achievable running disparity varies based on the tracking method used
    - DC-Balanced encoding must track the running disparity for both SpaceWire signals (Data and Strobe) simultaneously
      - The goal is to minimize the running disparity of each signal without unnecessarily minimizing one at the expense of the other
    - Modeling has shown that the 10-bit code set running disparity can be limited to eight or less for a bit stream composed of a random code sequence
  - Misses the run-length criterion by 40% (no mitigation method is available)
  - The ability to take advantage of the bandwidth efficiency of the 10-bit code set will depend on signal integrity analysis and experimentation

Summary

- An alternative character-level encoding method
  - Supports galvanic isolation of SpaceWire links using conventional AC-coupling circuits
  - Limits changes to the character-level of the standard
  - Provides transmission error detection comparable to the standard SpaceWire parity check
- The major tradeoffs to be considered
  - The impacts to SpaceWire link bandwidth efficiency
  - Encoding/decoding implementation complexity
  - Frequency performance using AC-coupled circuits
    - Needs further analysis and experimentation