New Approaches for DC Balanced SpaceWire

Session: SpaceWire Networks and Protocols, Short Paper

Alex Kisin
AS&D, Inc. work performed for NASA GSFC
Greenbelt, MD, USA
Alexander.B.Kisin@nasa.gov

Glenn Rakow
NASA Goddard Space Flight Center
Greenbelt, MD, USA
Glenn.P.Rakow@nasa.gov

Abstract—Direct Current (DC) line balanced SpaceWire is attractive for a number of reasons. Firstly, a DC line balanced interface provides the ability to isolate the physical layer with either a transformer or capacitor to achieve higher common mode voltage rejection and/or the complete galvanic isolation in the case of a transformer. Secondly, it provides the possibility to reduce the number of conductors and transceivers in the classical SpaceWire interface by half by eliminating the Strobe line. Depending on the modulator scheme – the clock data recovery frequency requirements may be only twice that of the transmit clock, or even match the transmit clock: depending on the Field Programmable Gate Array (FPGA) decoder design.

In this paper, several different implementation scenarios will be discussed. Two of these scenarios are backward compatible with the existing SpaceWire hardware standards except for changes at the character level. Three other scenarios, while decreasing by half the standard SpaceWire hardware components, will require changes at both the character and signal levels and work with fixed rates. Other scenarios with variable data rates will require an additional SpaceWire interface handshake initialization sequence.

Index Terms—SpaceWire, DC balance, Line encoding

I. INTRODUCTION

DC balanced data, where “0” and “1” ratio is 1 (or very close to 1) over a certain time stretch, allows data to go over capacitive or transformer barriers, thus creating better isolation for communication modules at different common ground potentials. Currently, these potential differences are a function of the common mode rejection of the receiver and, for Low Voltage Differential Signaling (LVDS), it is $+5/-4V$ at best. Originally, the SpaceWire hardware protocol was designed for an easy clock extraction and was not designed with DC balance in mind [3]. Over recent years there have been several attempts to create a DC balanced SpaceWire hardware protocol, but all of them either failed to create DC balanced Data and Strobe by an easy means [1], or rejected the Strobe line whatsoever thus forcing the user to extract a clock by using FPGA Phase Lock Loops (PLL) or using other techniques described in the Reference section of this paper [2]. Authors will try to review some new methods, both with and without a Strobe line being used.

II. METHODS WITH DATA AND STROBE LINES

II.A. Dual Complementary Bytes

One of the simplest methods will be splitting each data byte in to 2 bytes, where 1st byte is itself, along with Data Control Flag (DCF) and Parity (P), while 2nd byte is 1st byte inversion, including DCF and Parity, as seen in Fig. 1 below:

![Fig. 1. Original Byte Split](https://ntrs.nasa.gov/search.jsp?R=20160010303)

From a first glance it is obvious that Data line will be balanced for the full 20-bit sequence, but will it be true too for a Strobe? Likely, it can be easily shown that for any number of complementary bits divisible by 4, a Strobe will also be balanced: because 01 or 10 clock sequence always places its “0” or “1” “under” the same complementary data positions of both bytes – then Exclusive OR (XOR) results will also complement each other (Fig. 2 encircled color columns). Simulation shows that both Data and Strobe lines will be balanced for any bits combinations. It is also important to note that the parity bit is not a classical SpaceWire implementation – it is a parity of the 9 previous bits including DCF and data byte. And it is irrelevant whether Even or Odd parity is used for this method.

![Fig. 2. Data and Strobe DC Balancing Example](https://ntrs.nasa.gov/search.jsp?R=20160010303)

This scheme will also allow a “single error correction,” where the user can select either 1st or 2nd byte as the valid one, depending on which one’s parity was true. A maximum stretch of same bits sequence will be 18. If someone wants to reduce this stretch – they can try to play games of grouping bits and their complements between 2 bytes: like interleaving 2 adjacent bits with their complements will shrink the
maximum stretch to 6, but the error correction feature will be
gone.

Even we suggest to apply this same technique for all control
characters and time codes, using unbalanced control characters
and time codes everywhere except the initial handshaking
sequence will not significantly unbalance the Data and Strobe
lines and these characters can be used “as is” because of their
rare occurrences. However, during the handshaking sequence,
there is a possibility that a Null character, while being DC
balanced itself and its Strobe image is not, will be transmitted
by an Originator continuously when a Responder’s receiver is
not ready, thus charging the Strobe line and “saturating” the
LVDS receiver input beyond its common mode voltage
tolerances. To counter this problem, we suggest substituting
the original Null character of 01110100 with 10011100; as a
result, Strobe will be changed from 00100001 to 11001001.
Similarly, FCT character will be changed to 1100 and its
Strobe to 1001. As soon as the handshake phase is over – the
system can revert back to its original control characters.

The major drawback of this scheme is its half data rate.

II.B. PSEUDO RANDOM SEQUENCE (PRS) MODULATION

Another method partially described in [1] is using PRS
mixed with original data. It is easy to prove mathematically
that every meaningful data stream mixed on a bit by bit basis
with a random data stream becomes random itself.

Furthermore, every further XOR operation with this newly
minted random data will also produce random data. The PRS
organized on Linear Feedback Shift Registers (LFSR) [5] is a
close approximation for truly random data, and therefore,
can be counted as such, especially for longer generated sequences.
As a result, Data and Strobe created according to this might
also be considered random and thus DC balanced.

Note: all LFSR sequences do not contain combination when
all registers are equal to “0.” This creates a misbalance in “0”/
“1” ratio, because there can be a combination when all
registers are “1”. To solve this – we recommend to define an
LFSR state when it is 1 clock away from being all “1” and
then skip the all “1” state to the next consecutive state.

And yes, there can be unique situations described in [1]
when randomized Data or Strobe may have a long stretch of
the same bit values “0” or “1.” However, their probability is

extremely low, plus any resulting drift of hardware lines can be
mitigated by selecting LVDS receivers with wider
common-mode input voltage tolerances, such as Texas
Instruments product: SN55LVDS33-SP [4].

Initialization handshake is shown on Fig. 3 below. There
initialization Null and FCT characters should be selected by
the previously described DC balance criteria; afterwards a user
can revert back to using original control characters. It is also
important to note that while being disabled - LFSR’s first bit
mixed with data stream should be “0.” It is done to the PRS
initialization data sequence: LFSR is enabled after 1st cargo
“0” DCF is detected.

III. METHODS WITH DATA LINE ONLY

Removing the Strobe line is potentially a good idea: it will
increase wire bundle flexibility and reduce harness weight and
complexity as well as on-board electronic hardware.
Additionally, it also removes from SpaceWire its easy clock
extraction feature and makes its communication data rate
switching, as it is described in the original SpaceWire
protocol, more complex. However, taking advantage of
modern commercial and spaceflight FPGA’s features these
problems can be greatly alleviated.

III.A. DUAL NIBBLES WITH 4B/6B CODING [6]

It is suggested to substitute two of the original Data byte
nibbles with two 6-bit symbols. Each symbol will contain an
equal count of “0” and “1”: 3. Number of permutations for 3
“1” bits in 6-bit symbol for 64 symbols group is 20, which
means that each of 16 nibble’s combinations will be assigned
to its own DC balanced symbol, plus 4 extra symbols can be
used as 4 SpaceWire original Control characters.

No DCF bit will be required because Data and Control
characters are now unique, neither will be Parity bit: data
integrity will be checked by 3 “1” per symbol, or 6 per “byte.”

This method will probably provide an easiest DC balancing
implementation with only 20% of data bandwidth overhead.
Strobe can’t be used because it will not be DC balanced.

III.B. FIXED RATE WITH DUAL NIBBLES, BYTES OR PRS

The fixed rate with Dual Bytes or PRS schemes are selected
because they don’t require clock frequency switching. Clock

Fig. 3. Initialization Handshake Sequence with PRS

To be published in the proceedings for the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.
is extracted from an incoming Data stream using known FPGA PLL or Digital Lock Loop (DLL) techniques (see paper [2] References). Otherwise, these methods are basically the same as Data/Strobe PRS Modulation discussed in Section II.B.

III.C. VARIABLE RATES WITH DUAL NIBBLES, BYTES OR PRS

This method is also a derivative of the previous ones. Initial handshake at low rate will be done first and in a following cargo data Originator or Responder will notify each other about their desire to change data rate. After that, Originator shall break the existing link, wait for at least 6.4us (during which time both sides adjust and stabilize their clock generators) and repeat their handshake at a new rate as shown in Fig. 4 below.

IV. SUMMARY

This paper presented an incremental design approach option to improve SpaceWire, yet leverages most of the existing FPGA based SpaceWire designs for moderate data rate applications that require or may benefit from electrical isolation. It also describes an additional way to further reduce the mass and flexibility of the SpaceWire cables for applications that are tight on space. Additionally, it provides a means to specify a common physical layer and one which could work with any protocol that uses DC balanced line codes.

Table I below shows what are in author’s opinion brief characteristics of the above methods and some not covered additional ones. They might be a little biased, but nevertheless will provide a design engineer with possible guidelines.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Lines</th>
<th>RX Clock</th>
<th>Data Rate</th>
<th>Overhead vs. SpW</th>
<th>DC Balance</th>
<th>DC Quality</th>
<th>FPGA Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Standard SpaceWire</td>
<td>Data and Strobe (D&amp;S)</td>
<td>XOR-ed to Data and Strobe (D&amp;S)</td>
<td>Variable as is original SpaceWire</td>
<td>0%</td>
<td>Yes</td>
<td>Very good</td>
<td>Existing</td>
</tr>
<tr>
<td>2</td>
<td>Dual bytes (DB) encoding</td>
<td>Data only</td>
<td>4-phase sampling, or others</td>
<td>Fixed rate change requires dual handshaking</td>
<td>0%</td>
<td>Yes</td>
<td>Very good</td>
<td>Moderate=Complex</td>
</tr>
<tr>
<td>3</td>
<td>8b/10b (SWRI)</td>
<td></td>
<td></td>
<td></td>
<td>100%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate</td>
</tr>
<tr>
<td>4</td>
<td>16b/30b (SWRI)</td>
<td></td>
<td></td>
<td></td>
<td>50%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate=Complex</td>
</tr>
<tr>
<td>5</td>
<td>2 lines PRS modulation</td>
<td></td>
<td></td>
<td></td>
<td>0%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate</td>
</tr>
<tr>
<td>6</td>
<td>Fixed DB or PRS modulation</td>
<td></td>
<td></td>
<td></td>
<td>100%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate</td>
</tr>
<tr>
<td>7</td>
<td>Variable DB or PRS modulation</td>
<td></td>
<td></td>
<td></td>
<td>Same as in above DMS</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate</td>
</tr>
<tr>
<td>8</td>
<td>10b/10b or dual 4b/5b</td>
<td></td>
<td></td>
<td></td>
<td>0%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate=Complex</td>
</tr>
<tr>
<td>9</td>
<td>Manchester modulation</td>
<td></td>
<td></td>
<td></td>
<td>20%</td>
<td>Yes</td>
<td>Very good</td>
<td>Easy-Moderate</td>
</tr>
<tr>
<td>10</td>
<td>Dual nibble 4b/5b</td>
<td></td>
<td></td>
<td></td>
<td>0%</td>
<td>Yes</td>
<td>Good</td>
<td>Moderate</td>
</tr>
</tbody>
</table>

Table 1. Methods Comparison Chart

REFERENCES

To be published in the proceedings for the 7th International SpaceWire Conference, Yokohama, Japan, October 25-28, 2016.