Proceedings of the 9th International SpaceWire and SpaceFibre Conference 2022

ASI Agenzia Spaziale Italiana

esa

INGENIArs

STAR-Dundee

IEEE

AESS

Università di Pisa

SpaceWire and SpaceFibre 2022

Proceedings of the 9th International SpaceWire and SpaceFibre Conference

Pisa, Italy, 2022

Editors: Carole Carrie, Pietro Nannipieri and Steve Parkes

SpaceWire and SpaceFibre 2022

Proceedings of International SpaceWire and SpaceFibre Conference

Pisa, Italy, 2022

ISBN: 978-0-9954530-2-9

Preface

These proceedings contain the papers presented at the 9th International SpaceWire and SpaceFibre Conference, held in Pisa, Italy between 17th and 19th October 2022. This Conference brings together international spacecraft engineers and academics who are working on spacecraft on-board data-handling technology. It is of benefit to product designers, hardware engineers, software engineers, system developers and mission specialists, enabling them to share the latest ideas and developments related to SpaceWire and SpaceFibre spacecraft on-board network technologies.

SpaceWire is now being used or designed into well over one hundred spacecraft, covering science, exploration, Earth observation and commercial applications. High profile missions like James Webb Space Telescope, GAIA, ExoMars, BepiColombo, Sentinels 1, 2, 3 and 5 precursor, and GOES-R are using SpaceWire extensively. SpaceWire is being used in Europe, Japan, USA, Russia, China, India, and other countries of the World.

SpaceFibre is the next generation of SpaceWire technology, offering higher data-rates and substantially enhancing the capabilities of SpaceWire. It runs over electrical or fibre optic cable covering distances of 5m and 100 m respectively while running at lane speeds of up to 6.25 Gbit/s currently in radiation tolerant technology. The multi-lane link capability of SpaceFibre results in link speeds of 25 Gbit/s for a quad-lane link with up to 16 lanes per link being possible. Higher lane speeds are also possible. SpaceFibre is not only very fast, it incorporates quality of service, providing multiple independent virtual networks for transferring information over the physical network, each virtual network having its own priority, bandwidth allocation and schedule. These capabilities enable SpaceFibre to provide deterministic data delivery without loss of network bandwidth for combined control and payload data-handling networks. It also provides integrated, rapid fault detection, isolation and recovery technology, which makes SpaceFibre a highly robust network for use in applications where reliability and availability are critical.

The conference covers many different aspects of SpaceWire and SpaceFibre, and includes both academic and industrial presentations. Sessions cover standardisation, components, on-board equipment, test and verification, networks and protocols, and missions and applications. SpaceWire continues to be used extensively and SpaceFibre is gaining momentum, already being designed into spaceflight systems with the first missions in orbit. It is an exciting time in the SpaceWire community as this latest technology literally begins to take off.

The conference committee would like to acknowledge the support and hard work of the many individuals who made 9th International SpaceWire and SpaceFibre Conference possible. Originally planned for 2020 we are grateful that the global flood of pandemic has subsided sufficiently for the conference to go ahead in person in 2022. We appreciate the high-quality and inspiring contributions from the authors and the keynote speakers. We express our gratitude to the Technical Committee for their assistance in the review process. We recognise the support from the University of Pisa, IngeniArs, the European Space Agency and STAR-Dundee. Finally, we would like to give a special thanks to the conference organiser Carole Carrie (STAR-Dundee Ltd.), and the local organisers Pietro Nannipieri (University of Pisa) and Camilla Giunti (IngeniArs).

The Conference Chairpersons,
Luca Fanucci (University of Pisa),
Steve Parkes (STAR-Dundee Ltd.),
Felix Siegle (European Space Agency).

2022 Technical Committee

Barthelemy Attanasio – Thales Alenia Space, France

Thomas Bahls – DLR, Germany

Markus Bihler – DLR, Germany

Clifford Kimmery – USA

Daniele Davalle, Ingeniars Srl., Italy

Walter Errico: Sitael, Italy

Luca Fanucci, University of Pisa, Italy

Seisuke Fukuda – JAXA/ISAS, Japan

Hiroki Hihara – NEC, Japan

Jørgen Ilstad - ESA, The Netherlands

David Jameux - ESA, The Netherlands

Alexander Kisin – NASA GSFC, USA

Robert Klar - South West Research Institute, USA

Paolo Lombardi – Leonardo, Italy

Jim Lux - NASA JPL, USA

Giorgio Magistrati - ESA, The Netherlands

Keiichi Matsuzaki – JAXA, Japan

Silvia Moranti – ESA, The Netherlands

Pietro Nannipieri, University of Pisa, Italy

Masaharu Nomachi – University of Osaka, Japan

Steve Parkes – STAR-Dundee Ltd, UK

Paul Rastetter - Airbus GmbH, Germany

Krzysztof Romanowski – ITTI, Poland

Toru Sasaki – Mitsubishi Electric, Japan

Derek Schierlmann - Naval Research Laboratory, USA

Luca Serafini – Kayser Italia Srl., Italy

Felix Siegle - ESA, The Netherlands

Antonis Tavoularis – ESA, The Netherlands

Michael Walshe – Thales Alenia Space UK Ltd, UK

Programme Overview

Monday 17th October

09:00 – 16:30 Registration Open

14:00 – 16:00 Introduction & Key Note Speaker (120 min)

16:30 – 17:30 Networks & Protocols Short (60 min)

Tuesday 18th October

09:00 – 11:30 Registration Open

09:00 – 11:00 Networks & Protocols 1 Long (120 min)

11:30 – 13:00 Test & Verification Long (90 min)

14:00 – 15:00 Onboard Equipment Long (60 min)

15:00 – 16:00 Poster Session (60 min)

16:00 – 18:00 Components Long (120 min)

Wednesday 19th October

09:00 – 11:30 Registration Open

09:00 – 11:00 Components Short (120 min)

11:30 – 12:50 Test & Verification Short (80 min)

13:50 – 14:50 Networks & Protocols 2 Long (60 min)

15:20 – 16:40 Missions & Applications Short (80 min)

Programme is subject to change

Monday 17th October

Networks & Protocols (Short)

A Novel Encoder for DC-Balanced SpaceWire

Christopher M. Rose
Johns Hopkins University
Applied Physics Laboratory
Laurel, Maryland, USA
Christopher.Rose@jhuapl.edu

Steve S. Cho
Johns Hopkins University
Applied Physics Laboratory
Laurel, Maryland, USA
Steve.Cho@jhuapl.edu

Matthew M. Gile
Johns Hopkins University
Applied Physics Laboratory
Laurel, Maryland, USA
Matthel.Gile@jhuapl.edu

Kirk N. Volland
Johns Hopkins University
Applied Physics Laboratory
Laurel, Maryland, USA
Kirk.Volland@jhuapl.edu

Jarrett T. Wehle
Johns Hopkins University
Applied Physics Laboratory
Laurel, Maryland, USA
Jarrett.Wehle@jhuapl.edu

***Abstract*— A DC-balanced encoding for SpaceWire traffic offers several advantages in flight hardware design. Conventionally, block encoding techniques have been used. In this paper, a novel cycle-stretching method is described which takes advantage of SpaceWire's feature of embedding clock in data. This feature allows variable-length periods to be used opportunistically to perform DC-balancing of a SpaceWire stream, without using block encoders or relying on excursions from SpaceWire protocol. Such period-adjusting encoders can show high efficiencies for randomly generated SpaceWire traffic, particularly when paired with lookahead methods.**

An encoder whose output can be decoded by conventional SpaceWire receivers is presented. Such an encoder produces a data stream which, while being DC-balanced, can be analyzed by conventional test equipment (e.g. link analyzers) without additional hardware.

The efficiencies of multiple encoders are characterized. Several worst cases are considered. An encoder is tested with flightlike data from integration testing of DART (Double Asteroid Redirection Test) a NASA spacecraft which uses a SpaceWire network.

Keywords—SpaceWire, Encoder, Cycle-Stretching, DC-Balanced, AC-Coupled.

I. Introduction

Institutions that use SpaceWire possess a significant amount of SpaceWire-specific test, integration, and debug infrastructure. Not only hardware, such as link analyzers, data recorders, protocol checkers, but also supporting software, test procedures, and, significantly, the accumulated institutional knowledge.

The advantages of AC-coupled SpaceWire are well understood.[1,2] Prior papers have proposed encoding techniques for DC-balancing SpaceWire as an enabler for AC-coupling. Several block encoders have been offered.[3,4] Other approaches have included PRS modulation—combining SpaceWire signals with a pseudorandom sequence—and eliminating the strobe lines entirely.[5] All these proposed methods, with encoders that either break from the SpaceWire protocol,[6] or that otherwise obscure the packets' contents, undermine the value of existing infrastructure.

This is the problem our method was developed to address: to provide an encoder that is DC-balanced, without diverging from the SpaceWire protocol or rendering the traffic 'on the line' illegible. We hope to do so while maintaining an efficiency high enough to make the encoder competitive with conventional block encoding methods.

II. DC Balancing and Cycle Stetching

For an AC-coupled SpaceWire to work, the transmit streams on both sides of a link must each be DC-balanced; that is, for each signal, the number of 1's and 0's must be equal, considered over a 'long enough' interval. The cumulative difference between the number of 1's and 0's is called the 'cumulative disparity', or simply 'disparity'. We compute separate disparities for the data and for the strobe. These must always remain less than some figure we will call 'maximum disparity', whose value is dependent on features of underlying hardware.

The basic encoding of SpaceWire demands that either the data line or the strobe line must toggle between adjacent clock periods, but not both. In this way, the clock is encoded in the transmitted signals, and can be recovered by the receiving node without any explicit clock signal crossing the link. This reduces hardware and eliminates concerns about clock/data skew.

But this also offers an overlooked opportunity. A SpaceWire transmitter could delay transitions in both data and strobe lines simultaneously; in such event, no new data is transmitted.

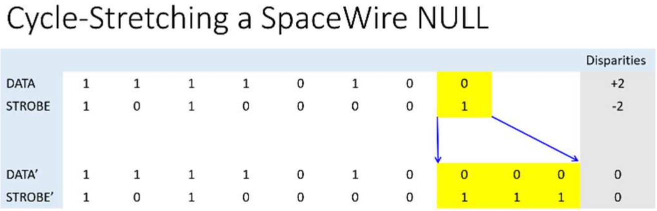
What if a node could be made to selectively delay transitions for the purpose of reducing the cumulative disparity on data and strobe signals? Such a node would pay a penalty in throughput, as the time of these additional transmit cycles will then be included, in which no new data is being transmitted.

Supported in part by a Janney Grant from Johns Hopkins University Applied Physics Lab, 2022.

A stream could be constructed to have a zero average disparity on both lines, data and strobe, and thus be ideal for AC-coupling. Also, such a SpaceWire signal would be interpreted at the receiver at as normal SpaceWire, adhering fully to the protocol, and with no additional decoding required. Further, the SpaceWire signals moving across the link would be readily interpretable by existing test and debug tools.

We have named this encoding method ‘cycle-stretching.’ A minimal implementation of such a method requires few logic gates beyond what is necessary for an unencoded SpaceWire node.

As an example, Figure 1 presents the bit pattern for a NULL token, the most common token on a SpaceWire link. On the upper rows, the data and strobe signals of the NULL are shown; in the right-hand column, we see that both the data and strobe signals have nonzero disparities. That is, there are two more 0’s than 1’s (assuming the parity for the prior character is zero) in the data bits, producing a +2 data disparity. There are two more 1’s than 0’s in the strobe bits, producing a net -2 strobe disparity.



Cycle-Stretching a SpaceWire NULL

											Disparities
DATA	1	1	1	1	0	1	0	0			+2
STROBE	1	0	1	0	0	0	0	1			-2
DATA’	1	1	1	1	0	1	0	0	0	0	0
STROBE’	1	0	1	0	0	0	0	1	1	1	0

Figure 1: Cycle-Stretching a SpaceWire NULL

At the level of a single NULL, these disparities may be tolerable by the hardware for an AC-coupled link. However, if a run of consecutive NULLs is sent, the cumulative disparities will grow linearly until the AC-coupled link fails.

The lower rows of Figure 1 show our cycle-stretch encoding. One cycle of data and strobe, conveniently, has values that can counterbalance the disparities in the character. The last cycle of the NULL, highlighted in yellow, has been stretched across three consecutive periods, producing a NULL that has zero disparities. The data conveyed on the data and strobe lines, however, is not changed; the receiving node just sees a NULL. It takes 25% longer to be transmitted.

It is a little suspicious how this feature has remained latent in the protocol this whole time. It is as if the inventors of SpaceWire anticipated our need, and embedded this ingenious degree of freedom in the protocol itself, just for us to find it.

III. The Split Difference Method

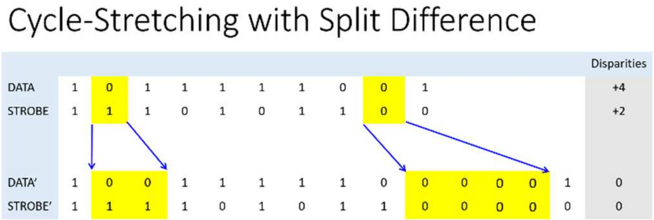
Not all examples work out so conveniently as our NULL above. Often the magnitude of the data and strobe disparities will be different. In these cases, we can zero both disparities within the same character, by stretching two different data/strobe pairs a differing number of cycles.

We call this the split difference method. If there are two different disparities, D_m and D_n, where D_m is the disparity with the larger absolute magnitude, then we would perform stretches with different numbers of cycles:

$$\text{Major stretch} = D_n + ½(D_m – D_n) \qquad (1)$$

$$\text{Minor stretch} = ½(D_m – D_n) \qquad (2)$$

Figure 2 shows an example that will make this clear.



Cycle-Stretching with Split Difference

															Disparities
DATA	1	0	1	1	1	1	1	0	0	1					+4
STROBE	1	1	1	0	1	0	1	1	0	0					+2
DATA’	1	0	0	1	1	1	1	1	0	0	0	0	0	1	0
STROBE’	1	1	1	1	0	1	0	1	1	0	0	0	0	0	0

Figure 2: An Example of the Split Difference Method

In this case, $D_m = 4$, and $D_n = 2$. As the expressions above indicate, the major stretch should be three cycles (or periods) long, and the minor stretch should be one.

There are additional details to consider to implement a full DC-balanced encoder using cycle-stretching, even in this simplest mode—with each character being individually balanced. Not all four combinations of data and strobe (0/1, 0/1, 1/1, and 1/0) may occur in some characters; in these cases, the disparity of the unstretched character should be added to that of the subsequent character. Timecodes must be free of any cycle stretching, as it is a requirement to transmit them with minimum delay. An upper limit to the number of consecutive stretch cycles must also be chosen and enforced.

IV. Worst Cases and Efficiencies

The largest cost for per character DC-balancing via cycle stretching is the reduction in net transmission rate, and accompanying reduced ability to predict the real data rate, of a cycle-stretched link.

SpaceWire data characters, which are by far the most common in SpaceWire traffic, are inherently unbalanced.

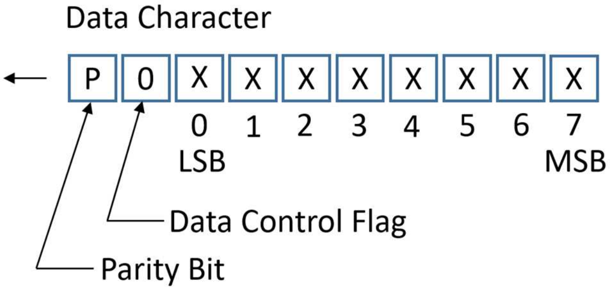


Figure 3: Structure of a SpaceWire Data Characters

For a data character, the data-control flag must be zero. Also, the parity bit is given as even; but, for a ten-bit character, it is not possible to compose a character with an equal number of 1’s and 0’s that also has an even parity (i.e. ten bits, balanced as five 1’s and five 0’s, can only have odd parity.) Therefore some amount of cycle-stretching will always be necessary.

For a conventional block encoder, the resulting data rate reduction is easily quantifiable regardless of the input stream. A block encoder can provide a consistent data rate because the worst-case is enforced everywhere; that is, the same number of bits are added to each token. For instance, for an 8b/10b encoder, every eight bit input character produces ten bits of encoded data. In contrast, a cycle-stretch encoder can produce a range of efficiencies, dependent on statistical qualities of the data to be encoded.

For cycle-stretch encoders, the fraction of bandwidth consumed by cycle-stretching is variable, and dependent on the contents of the input stream. The method takes advantage of input streams that are well-balanced, but must use more transmit cycles to encode input streams that have large disparities.

The simplest application of cycle-stretching for DC-balancing would perform cycle-stretching in each character to arrive at zero or near-zero disparities. Such an encoder was implemented by the authors by modifying SpaceWire node IP written at GSFC (Goddard Space Flight Center) by Glenn Rakow et .al. The node was incorporated into a ten-port router similar to those used in the avionics for Parker Solar Probe and IMAP (Interstellar Mapping and Acceleration Probe), both NASA missions built by Johns Hopkins University Applied Physics Laboratory in Laurel, Maryland. The design was implemented on a Microchip ProASIC 3000.

For long runs of worst-case data, this simple cycle-stretch encoder could approach a Manchester encoder in its efficiency. Consistent worst-case or near-worst-case input data would nearly double the bandwidth required to transmit each packet, if the minimal interval between transitions were kept unchanged.

Figure 4 below shows an oscilloscope image of the traffic on the router, with the router's cycle-stretch encoders enabled. It doesn't require a very sophisticated eye to see that, for the simple implementation, the added 'stretch' cycles significantly reduce the efficiency of the encoders. However, and notably, the downstream receivers for these packets, connected avionics hardware or GSE test equipment, interpreted them with no errors, without any additional decode logic required.

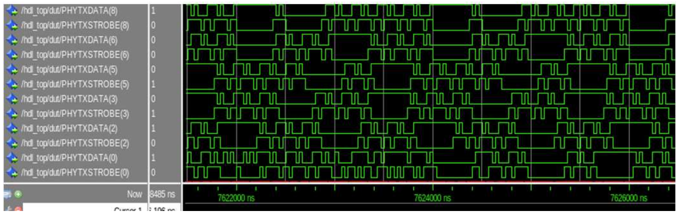

/hdl_top/dut/PHYTXDATA(8) 1
/hdl_top/dut/PHYTXSTROBE(8) 0
/hdl_top/dut/PHYTXDATA(6) 0
/hdl_top/dut/PHYTXSTROBE(6) 0
/hdl_top/dut/PHYTXDATA(5) 0
/hdl_top/dut/PHYTXSTROBE(5) 1
/hdl_top/dut/PHYTXDATA(3) 0
/hdl_top/dut/PHYTXSTROBE(3) 1
/hdl_top/dut/PHYTXDATA(2) 1
/hdl_top/dut/PHYTXSTROBE(2) 0
/hdl_top/dut/PHYTXDATA(0) 1
/hdl_top/dut/PHYTXSTROBE(0) 0
Now 8485 ns
7622000 ns
7624000 ns
7626000 ns

Figure 4: Traffic on a Simple Cycle-Stretching Router Implementation

We measure the efficiency of an encoding method by calculating the number cycles of input data to be encoded divided by the resulting number of encoded cycles. For a stream of generated packets of random length, with each data bit randomly selected with 0's and 1's being equally likely, the resulting efficiency of this simple cycle-stretch encoder is approximately 0.66. This simplest application of cycle-stretching would be appropriate only where there was significant available bandwidth overhead.

One can easily compose worst cases for cycle-stretch encoding, with either the transmit data or strobe signal containing an arbitrarily long series of either ones or zeroes. While these worst cases are bad outcomes for the encoder efficiency, they are unlikely to occur in practice. A packet of all zeros is uninteresting because it conveys little information. In such a worst-case situation, an encoder could insert NULL characters interior to packets, or between packets, to create opportunities to perform cycle-stretching and reduce disparities.

Another worst-case exists: a case where the input data to be encoded is composed entirely of timecodes. Nothing in the SpaceWire protocol prevents this scenario. Obviously, an input stream that is nothing but back-to-back timecodes would foil any cycle-stretching method that left timecodes unaltered. In practice, such a usage could be proscribed a priori.

A SpaceWire node could be modified to improve its efficiency while using cycle-stretch encoding. It could compress the data to be transmitted, or cherry-pick only interesting data to be sent. It could combine incoming data with some easily derived sequence, for instance one produced by an LFSR, (as suggested by Kisin and Rakow, [4]) to improve the amount of activity. All these actions, however, make interpretation of packet traffic, for instance through a link analyzer, more difficult. Making packets less legible 'on the line' is counter to the aims of this project.

V. Lookahead Methods

Most flight-like SpaceWire data steams possess, to varying degrees, the tendency of reversion to the mean. The data to be sent, over a sufficient time period, averages to roughly half 1's and half 0's. This feature can be exploited.

A more efficient cycle-stretch algorithm, with the ability to see a fixed number of upstream characters (e.g. characters to be sent in the future), and a tolerance for disparities below a specified threshold, could choose to defer some balancing tasks. It could wait, opportunistically, allowing some positive and negative disparities to cancel one another, reducing the total number of stretch-cycles necessary. Such an encoder would, ideally, employ cycle-stretching only to correct disparities as they approached the threshold, that is, the maximum tolerable disparity.

We have investigated using such look-ahead methods to reduce the amount of cycle-stretching required.

Two parameters are used. One is the number of periods ahead the algorithm can see in its decision-making, called 'lookahead distance'; the second is the maximum disparity to be allowed.

The authors wrote a C language model to demonstrate the impact on encoder efficiency of these features: lookahead method, lookahead distance, maximum allowable disparity. The C code model is a (somewhat) rapid tool for estimating an algorithm's efficiency for a given data stream. Additionally it can produce randomized input data as well as that of some cases of specific interest. The authors hope to be able to provide this model to interested readers.

In our first method, what we have called ‘naive lookahead,’ the difference in the accumulated disparities between the lookahead character and the current character to be transmitted is divided by the lookahead distance (which is the number of periods separating them in the transmit character sequence.) The rounded result is then the current disparity to be zeroed via cycle-stretching in the current character. This calculation is repeated twice, to find the current disparity for the data and for the strobe. The decimal fractions of disparities that are rounded down are not discarded, but added to the differences for the next cycle’s calculations.

Some things to note about naive lookahead: it is likely to be inefficient for input data that undergoes sudden statistical changes. Also, because it uses division, it is subject to errors in accuracy when implemented in any finite digital hardware. It requires some exception cases for the beginning and end of any finite run of input data.

A second method we attempted uses a ‘rolling average’ in a similar fashion. For each new character to be transmitted, we would add the lookahead character’s disparity, and subtract the current outgoing character’s disparity, to a rolling average. This rolling average then would be divided by the lookahead length, to derive the current disparity to be balanced. This same calculation would be employed for data and for strobe, to produce two disparities.

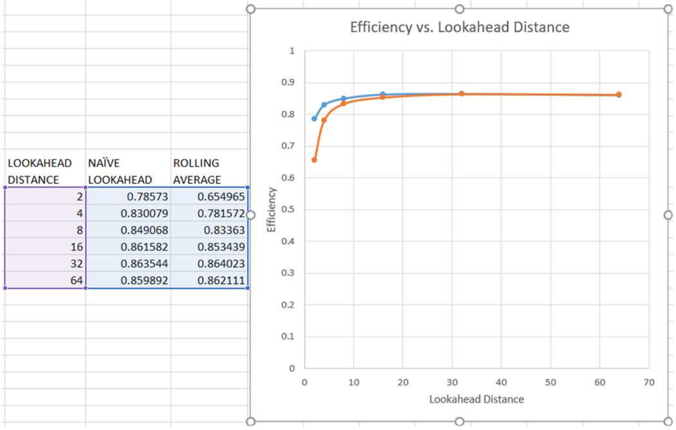
Both of these lookahead methods would require significantly more logic and memory to implement. In both of these methods, the accumulated disparities for both the lookahead character and the current (outgoing) character would need to be calculated. Obviously, encoder latency will be increased in naive lookahead and in rolling averaging methods.

Initial results were not encouraging. The efficiencies were higher than for the simple cycle-stretching method, but the resulting disparities increased over time.

Both of the lookahead methods are inherently ‘leaky’ in that they use averaging and division. To ensure that DC-balancing is maintained, a ‘clean-up’ stage was added to process the output of the lookahead stage. This clean-up stage would perform simple single-character disparity zeroization only for characters whose disparities exceeded a maximum allowable disparity.

Subsequent to the addition of this second stage, the efficiencies were higher and the disparities were well-behaved.

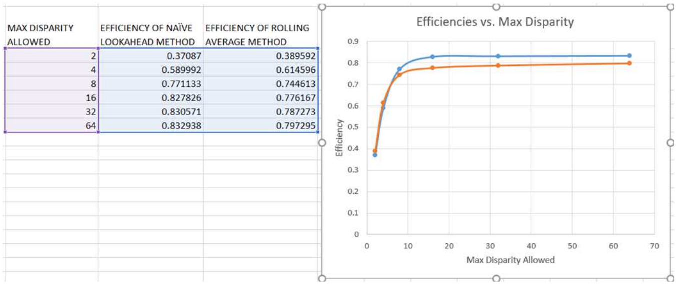
In one series of simulations, the encoding of the same long sequence of randomly-generated data was performed iteratively with a larger lookahead value each time. Figure 5, below, captures the results, showing that a larger lookahead distance, for the same given disparity thresholds, results in higher efficiency.



LOOKAHEAD DISTANCE	NAÏVE LOOKAHEAD	ROLLING AVERAGE
2	0.78573	0.654965
4	0.830079	0.781572
8	0.849068	0.83363
16	0.861582	0.853439
32	0.863544	0.864023
64	0.859892	0.862111

Figure 5: Graph of Efficiency versus Lookahead Distance

In a second test, the same fixed lookahead value was used, while the simulation was iterated with ascending values of the disparity thresholds—the encoder will ignore higher fluctuations in the instantaneous disparities so long as they remain below the thresholds. Figure 6 shows, not surprisingly, the higher thresholds resulted in reduced amounts of cycle-stretching performed, and thus higher efficiencies.



MAX DISPARITY ALLOWED	EFFICIENCY OF NAÏVE LOOKAHEAD METHOD	EFFICIENCY OF ROLLING AVERAGE METHOD
2	0.37087	0.389592
4	0.589992	0.614596
8	0.771133	0.744613
16	0.827826	0.776167
32	0.830571	0.787273
64	0.832938	0.797295

Figure 6: Graph of Efficiency versus Max Disparity

For sufficiently random data, it’s clear that the naïve lookahead method is sufficient for a high efficiency, with a reasonable lookahead distance and disparity threshhold. For less random data, the rolling averages method proved more consistently successful.

Additionally, lookahead simulations were performed using data recorded from integration tests of the DART spacecraft. DART, the Double Asteroid Redirection Test, is a recent NASA mission undertaken by JHUAPL to demonstrate planetary defense. Several runs of different samples of data produced a range of efficiencies, from dismaying (0.721749, for data with a lot of zeroes) to gratifying (0.822454). These results were produced with a rolling average, a lookahead distance of 16, and a max allowed disparity of 32.

As a final note: in any flightlike SpaceWire traffic, there are quiescent periods with no packets being transmitted. Such intervals of idleness can by used to reduce or eliminate disparities, by performing cycle-stretching over the series of NULL characters that must be sent during the gaps in traffic. The above efficiency discussions assume that all packets are back-to-back, without any quiescent periods.

VI. Future Work and Applications

More work remains in developing an understanding of how to find the maximum allowable disparity is for a given piece of SpaceWire hardware. The abilty to measure that value accurately is crucial to the performance of any cycle-stretching encoder.

Also, existing algorithms modeled in C need to be converted to syntheziable HDL so that the resources required to implement them can be quantified.

It's clear, also, that there are better cycle-stretch algorithms waiting to be discovered. For any given sequence of characters, there must exist one or several optimum cycle-stretch sequences that will produce a minimal-length DC-balanced sequence, and we currently have no idea how to find them.

AC-coupled SpaceWire offers a higher degree of robustness particularly when charging spacecraft internal and external components occurs, for instance due to immersion in plasmas. This is of particular concern in deep space missions. For this reason, AC-coupled SpaceWire, using a cycle-stretch encoding, has been baselined for the proposed Interstellar Probe mission.[7] It is our goal to have a viable, robust, and efficient encoder ready when called on for that mission.

References

[1] M. Suess, J. Ilstad, W. Gasti, "Galvanic Isolated SpaceWire Links, Requirements, Design Options and Limitations," 2009, ESA Workshop on Reliable Power & Signal Interfaces

[2] [2] M. Epperly and S. Torno, "Galvanically Isolated SpaceWire," International SpaceWire Conference 2014

[3] [3] Cliff Kimmery, "DC-Balanced Character Encoding for SpaceWire," International SpaceWire Conference 2011

[4] [4] M. Epperly and S. Torno, Ibid.

[5] [5] A. Kisin and Glenn Rakow, "New Approaches for Direct Current (DC) Balanced SpaceWire," 7th International SpaceWire Conference, 2016.

[6] [6] "SpaceWire – Links, Nodes, and Networks", ECSS Standard ECSS-E-ST-50-12C

[7] [7] Kinnison et. al. "Conciderations for a Pragmatic Interstellar Probe Mission,", December 2020, American Geophysical Union, Fall Meeting 2020.

Evolutions of SpaceWire Protocols for Deterministic Data Delivery

Networks and Protocols, Short Paper

Krzysztof Romanowski*, Piotr Tyczka
ITTI Sp. z o.o.
Poznań, Poland
{Krzysztof.Romanowski,Piotr.Tyczka}@itti.com.pl

David Jameux
ESTEC
European Space Agency
Noordwijk, The Netherlands
David.Jameux@esa.int

***Abstract*—The primary application domain of SpaceWire networks has traditionally been payload data handling, whereas communication related to command and control has been implemented with other types of networks, since the basic SpaceWire protocol cannot provide deterministic data delivery. There have been several protocols proposed that add determinism to SpaceWire, including SpaceWire-RT, SpaceWire-T, and SpaceWire-D. This paper presents further evolutions in this area based on SpaceWire-D as the starting point. A new protocol called Mixed Criticality Message Passing protocol (MCMP) is proposed, together with a new intermediate protocol for transferring data in time slots instead of RMAP, called the MCMP Register Access Protocol. The protocol has been implemented and tested in the MOST-X network simulator.**

Keywords—network protocols, SpaceWire, deterministic data delivery, mixed criticality systems, simulation

I. Introduction

Adopting a single common network for all on-board communication can lead to reduction of complexity, mass, and power demand. However, the timeliness requirements of command and control applications call for deterministic data delivery with guarantees on latency and jitter that are absent in the basic SpaceWire protocol [1].

Several protocols have been proposed that add determinism to SpaceWire, including SpaceWire-RT [2], SpaceWire-T [3], and notably SpaceWire-D [4]. The latter schedules SpaceWire traffic, defining four virtual buses of varying classes of determinism and multiplexing the traffic on them in the time domain, with time slots defined by SpaceWire time-codes sent by a network manager. The Remote Memory Access Protocol [5] is used as the transport mechanism between network nodes.

This paper describes evolutions in this area that were proposed in the ESA-funded project *SpaceDet*. Taking SpaceWire-D as the starting point, a new protocol called Mixed Criticality Message Passing protocol (MCMP), is introduced, together with a new intermediate protocol for transferring data in time slots instead of RMAP, called the MCMP Register Access Protocol (MRAP).

Section II presents background information, including basic features of SpaceWire-D and some limitations. The core components of the proposed MCMP protocol are the subject of Sections III, IV, and V. Testing and validation are discussed in Section VI, and Section VII presents conclusions.

II. Background

The SpaceWire-D protocol schedules SpaceWire traffic, defining four virtual buses of varying classes of determinism and multiplexing the traffic on them in the time domain, with time slots defined by SpaceWire time codes sent by a network manager. While the protocol seems very well suited for traffic comprising RMAP transactions of not very long packets in networks of moderate complexity, there are some characteristics of SpaceWire-D for which it is worthwhile to analyse their suitability for less typical use cases and to investigate possible alternative solutions.

The mechanism used in SpaceWire-D for defining the time-slots is the SpaceWire time-codes, which can – according to the SpaceWire-D specification – be used either:

- directly: each time code received by a SpaceWire-D initiator defines the beginning of a time-slot ([4], Clause 5.5.2.1); or
- indirectly: time-slots are defined by a local timer in the SpaceWire-D initiator, with time codes used for synchronizing the slots; a time code that arrives too early or too late (i.e. beyond specific limits) with respect to the time-slot boundary based on the local timer causes the timer to update its time and correct the time-slot boundary immediately ([4], Clause 5.5.2.3).

Although the latter tolerates occasionally missing time codes, in both methods the time codes are expected to be generated at each time-slot boundary. The frequency of the codes is thus related to the needs of the synchronous control applications that require the deterministic protocol. These are typically used in control loops of frequencies of 1 Hz to 1 kHz. For such a range of frequencies, and in addition for possibly less frequent large scientific data transfers, all running on the same network, the mere 64 time-slots might be not flexible enough.

All the traffic under SpaceWire-D is carried as RMAP transactions. This has the advantage of RMAP being already implemented in a number of devices, which – when acting as SpaceWire-D (and RMAP) target nodes – basically do not require any additional protocol support, the whole burden of SpaceWire-D implementation involving only the initiator nodes. Using RMAP is naturally convenient when the traffic intended to be sent via SpaceWire-D is composed of transactions (of a command-reply type), and of RMAP transactions in particular.

However, such use might potentially conflict with regular RMAP applications as far as memory addressing space is

This work has been funded by the European Space Agency under contract no. 4000125047/18/NL/BJ/zk.

**Corresponding author*

concerned. Also, the details of using RMAP in SpaceWire-D are not necessarily compatible with regular RMAP usage, the most evident example being probably the extra word added to the RMAP payload on the SpaceWire-D Packet Bus due to possible segmentation. On the other hand, the overhead of all the data in RMAP headers may be more than is needed for the deterministic transmission, and even the transactional character with the acknowledgements may not always be required.

Finally, fault detection, isolation, and recovery (FDIR) is a valid aspect for any communications protocol, and particularly in applications where deterministic transmission is expected. The SpaceWire-D draft standard addresses fault detection (and prevention; [4], Clause 5.16), but not isolation or recovery. This is a potential area for improvement.

As a possible means for the improvement, modifications and additions to SpaceWire-D are proposed that can be classified as three areas: time coordination, intermediate protocol, and FDIR mechanisms. The name for the resulting modified protocol is MCMP. Time coordination

MCMP uses local timers for time-slot definition and a means of synchronising the SpW Nodes local timers – e.g. the SpaceWire Time Distribution and Synchronization Protocol (SpW-TDSP) [6].

Due to the flexibility of structuring the epoch, it is possible to define long epochs and short time-slots, for use cases involving both slow and fast periodic transfers. IN cases where long time-slots are useful (for transmission of large data packets), the SpaceWire-D mechanism of multi-slots can be used. There is also a possibility of using non-uniform time-slot sizes, with the effect basically similar to that of using multi-slots, but with the definition of non-uniform sizes being permanent for a given system and common for all masters[1] rather than being created ad hoc by a specific master opening a multi-slot virtual bus. Short time-slots can be appropriate for time-critical traffic on static and dynamic buses, while long slots can be useful for high volume data on asynchronous and packet buses.

The following principles apply to MCMP:

1. Local timers are mandatory at master nodes and optional at slave nodes. Slave nodes respond to master nodes and do not themselves initiate transactions, thus they do not need to be aware of time, time-slots, and schedules. It is the responsibility of master nodes to maintain the time-slot sequence and obey the schedules. However, slaves are required to close master-initiated transactions within a bounded time known to the master(s).

2. The time-slot sequence needs to be the same at all master nodes. Since time codes are no longer used to define the time-slots, the slots have to be identified by relation to local timer values of their beginnings, relative to the beginning of the epoch. The beginning and length of the epoch need to be consistent among the master nodes and should be set at system initialization time.

3. The schedule tables are maintained by each master node and manipulated by opening and closing virtual buses. The tables may be the same at each master node (simple scheduling) or they may be different (concurrent scheduling), provided there is no conflict in the nodes' usage of network resources (ports/links). Assuring there is no such conflict is the responsibility of applications manipulating the tables and is a functionality outside the scope of MCMP.

Synchronization of the local timers of nodes is the responsibility of SpW-TDSP (or of any other time synchronisation means) and is beyond the scope of MCMP. However, due to the way MCMP works, there are some requirements and notes:

a. A master node is not allowed to initiate any transaction until its local timer has been synchronized, including any latency, jitter, and drift mitigation that may be applied – unless the node is also the SpW-TDSP initiator (or unless there is only one SpW-MCMP master).

b. The messages of any SpW-TDSP can only be channelled through a specific virtual bus of MCMP. It may be adopted that at least the first MCMP static bus, allocated to the first time-slot in an epoch, is dedicated to SpW-TDSP transactions; and that the SpW-TDSP initiator runs in an MCMP master node and needs to always have information on the epoch origin and length, and at least on its dedicated time-slot(s).

c. MCMP masters have also the role of MCMP slaves when receiving time data from the SpW-TDSP initiator.

Fig. 1 shows the different elements of the MCMP functionality in a stack of protocols layers.

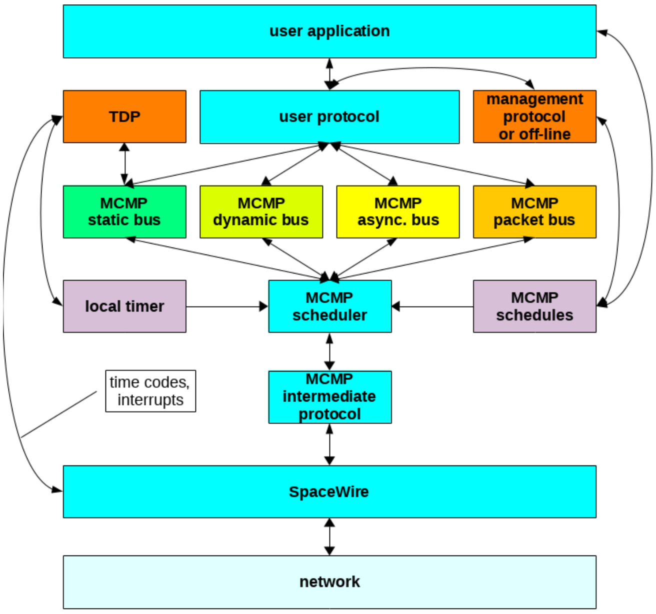


Fig. 1. Stack of protocols engaged in MCMP operation

III. Intermediate Protocol

It is proposed that solutions alternative to RMAP be allowed for the "intermediate" protocol between user data units and SpaceWire packets.

At the upper layer of this functionality, somewhat related to scheduling, is the transaction model. The following two models – mutually exclusive -- have been proposed for MCMP:

[1] The notions of *master* and *slave* are used in MCMP in place of the roles of *initiator* and *target* in SpaceWire-D.

1. [Primary model] “Acknowledged” – as it is in SpaceWire-D, where a complete transaction, scheduled in a slot, has to include both the command and the reply (and so RMAP write commands should set the ‘reply’ bit in the command field in the RMAP header; for RMAP read commands this is clearly the only option). The acknowledgement can be generated also for other protocols, and there need not be a one-to-one relation between the command and the reply packets (cf. aggregated acknowledgements in SpaceWire-T). The advantage to this model is the possibility of detecting transfer faults (e.g. loss of the command or the reply packet) at the master node.

2. [Secondary optional model] “Unacknowledged” – where the operation scheduled in a slot includes only one-way communication. This can be e.g. an RMAP write command with the ‘reply’ bit unset or a write-like command of some other protocol. This model is currently not within the SpaceWire-D specification.

Going down in the protocol stack, the intermediate layer uses some means of passing user data to the SpaceWire layer. The following two mutually exclusive solutions encapsulating the user data have been proposed for MCMP:

a. [Primary solution] A dedicated (with its own protocol ID) protocol based on RMAP.

While keeping the original RMAP header structure, the semantics of some of the fields can be customized for the new protocol. If some limits are placed on the RMAP functionality and address range, certain fields or bytes in the RMAP header can be freed for other use. For example, enough space in the header could be found for a long time-slot.

Specifically, the memory address field, no longer (or not necessarily) related to the same memory addresses as in RMAP, can also have a different semantics (addressing some registers of a special set) and leave some bits free. If the extended address is not used (e.g. because 40-bit address space is not needed and it is enough to be able to address 2^{32} memory units), then 8 bits are freed for other use.

Altogether, the above change can provide as much as 24 bits of space for information specific to MCMP. It is proposed that this specific information includes the time-slot identifier. The SpaceWire-D protocol makes use of 64 time-slots, which can be identified with a 6-bit-wide field. Adopting SpW-TDSP for MCMP gives a possibility for a much larger number of time-slots. It is assumed that there are cases where having more than 256 time-slots in an epoch may be useful, so that a single byte for the time-slot ID is not enough and a 2-byte field should rather be adopted. The two bytes can occupy the place of the ‘extended address’ field and one of the ‘data length’ bytes of the RMAP header format.

An example packet header format for the proposed MCMP is shown in Fig. 2.

Besides changes in the packet format, some of the RMAP command variants (e.g. the Read-Modify-Write) can prove unnecessary in MCMP, making a simplification of the protocol possible.

This RMAP-derived intermediate protocol for MCMP has been termed the MRAP (MCMP Register Access Protocol).

b. [Secondary alternative solution] Raw transmission: no encapsulation at all. In this variant, the user data unit is transmitted as is in an appropriate time-slot. The MCMP in this case can be seen as the SpaceWire-D protocol reduced to its upper layer of scheduling (Fig. 3).

IV. FDIR MECHANISMS

The current specification of SpaceWire-D provides means of detecting some faults: time-code errors (early, late, or missing), SpaceWire errors (link connect failures, reception of EEP), errors signalled in RMAP replies, and late or missing RMAP replies. These capabilities are extended in the proposed MCMP protocol with the following mechanisms.

1. Time-slot identification – already present in the current SpaceWire-D as “virtual bus ID” (concatenated with other parameters in the transaction ID field of the RMAP header). It is also present in the MRAP protocol. This ID makes it possible to match replies against commands, and to check whether the packet arrives in the proper time-slot.

2. SpaceWire interrupts. In a scenario where no multi-slots are employed, the interrupts are used for “hard limitation” of the time-slots, by forcing the routing switch to discard any packets that are being transferred at the time of a time-slot boundary.

3. Guard (silent) intervals defined to be in each time-slot or multi-slot. Allocated at the beginning and at the end of a time-slot, they can accommodate time-slot time skew and jitter.

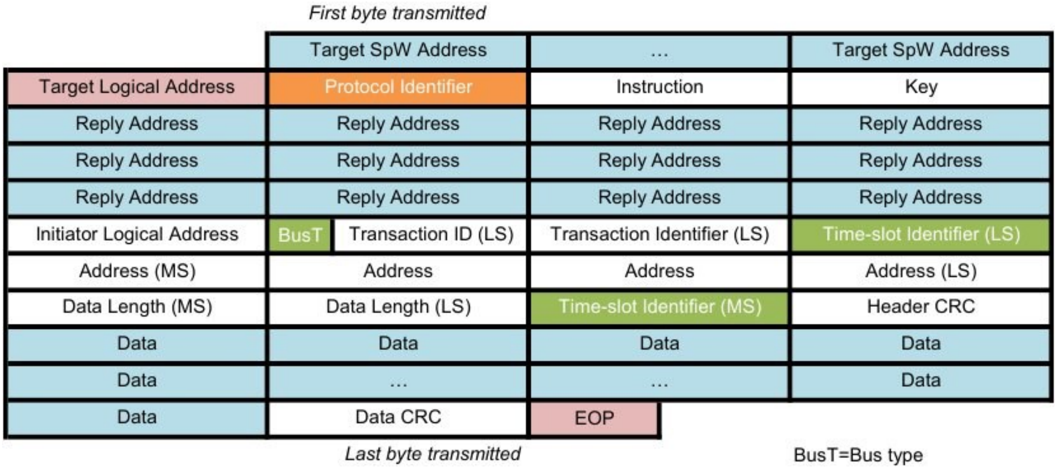
V. TESTING AND VALIDATION OF MCMP

As a simulation environment the MOST-X simulator developed by Thales Alenia Space [7] based on ns-3 [8] was selected. The simulator was substantially extended so as to make it possible to be used for simulations of SpaceWire-D as well as its evolutions – the MCMP protocol.

The network modelled in the simulator consisted of different variants of connections of RMAP nodes and switches; a representative example is shown in Fig. 4. The switch models were equipped with the ability to model a feature of the Cobham Gaisler GR718B switch of truncating packets on reception of time-codes or distributed interrupts, which could be used in support of SpaceWire-D protocol or its evolutions.

A number of data streams were defined, following the SAVOIR classes, and scheduled to use all types of buses: static, dynamic, asynchronous, and packet. Various time epochs were used, with time slot numbers ranging from 64 to 8000.

The tests showed that the simulator application is capable of simulating RMAP and SpaceWire-D in basic use cases, also with all the four SpaceWire-D bus types co-existing in the same simulation run. The latencies and throughputs recorded were as expected, given the assumed SpaceWire link data rate and the device latencies (which were taken to be similar to those reported in documentation of some SpaceWire devices available on the market, e.g. the SpW-10X routing switch.



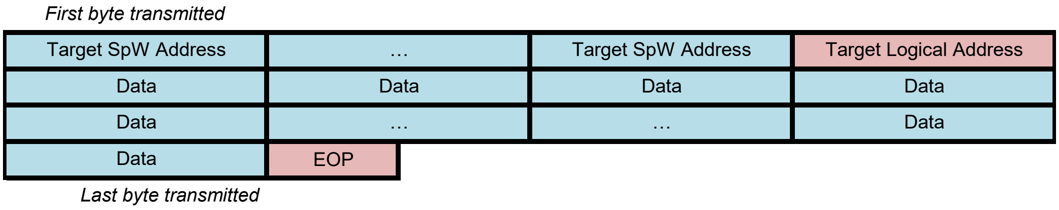
First byte transmitted

	Target SpW Address	…	Target SpW Address
Target Logical Address	Protocol Identifier	Instruction	Key
Reply Address	Reply Address	Reply Address	Reply Address
Reply Address	Reply Address	Reply Address	Reply Address
Reply Address	Reply Address	Reply Address	Reply Address
Initiator Logical Address	BusT Transaction ID (LS)	Transaction Identifier (LS)	Time-slot Identifier (LS)
Address (MS)	Address	Address	Address (LS)
Data Length (MS)	Data Length (LS)	Time-slot Identifier (MS)	Header CRC
Data	Data	Data	Data
Data	…	…	Data
Data	Data CRC	EOP	

Last byte transmitted

BusT=Bus type

Fig. 2. Fields of an MRAP write command



First byte transmitted

Target SpW Address	…	Target SpW Address	Target Logical Address
Data	Data	Data	Data
Data	…	…	Data
Data	EOP		

Last byte transmitted

Fig. 3. Unencapsulated packet of arbitrary SpaceWire protocol

VI. Conclusion

The general conclusion that can be drawn from the simulation results is that the new proposed MCMP protocol operates correctly and assures deterministic data delivery, which satisfies the fundamental prerequisite for it while simultaneously offers more flexibility in adapting to different network traffic types. Further work is needed to evaluate the protocol in a non-simulated environment and to compare this approach of placing most of the responsibility for deterministic transport on end nodes versus the alternative solution of placing that responsibility mostly on switches.

References

[1] European Cooperation for Space Standardization, “SpaceWire – Links, nodes, routers and networks,” ECSS-E-ST-50-12C Rev.1. Noordwijk: ECSS Secretariat, 2019.

[2] S. Parkes, “SpaceNet – SpaceWire-RT Initial Protocol Definition,” Draft A Issue 2.1, University of Dundee, 2008.

[3] S. Parkes and A. Ferrer Florit, “SpaceNet – SpaceWire-T Initial Protocol Definition,” Draft A Issue 3.1, University of Dundee, 2009.

[4] S. Parkes, A. Ferrer, and D. Gibson, “SpaceWire-D Standard,” Draft E, Issue 0.4, University of Dundee, 2015.

[5] European Cooperation for Space Standardization, “SpaceWire – Remote memory access protocol,” ECSS-E-ST-50-52C. Noordwijk: ECSS Secretariat, 2010.

[6] S. Habinc, A. Sakthivel, and M. Suess, “SpaceWire – Time Distribution Protocol,” Proc. 5th Int. SpaceWire Conf. Gothenburg 2013, pp.363-367.

[7] B. Dellandrea and D. Jameux, “MOST: Modeling of SpaceWire Traffic,” Proc. 5th Int. SpaceWire Conf. Gothenburg 2013, pp. 281-285.

[8] https://www.nsnam.org

SPACEMAN 2: Towards Multi-Protocol Network Management

Networks and Protocols, Short Paper

Jarosław Kwiatkowski, Krzysztof Romanowski*, Piotr Tyczka
ITTI Sp. z o.o.
Poznań, Poland
{Jaroslaw.Kwiatkowski,Krzysztof.Romanowski,Piotr.Tyczka}
@itti.com.pl

Rafał Renk
Adam Mickiewicz University
and ITTI Sp z o.o.
Poznań, Poland
Rafal.Renk@amu.edu.pl

David Jameux
ESTEC
European Space Agency
Noordwijk, The Netherlands
David.Jameux@esa.int

Abstract—**Initiatives are on-going to upgrade the current communication technologies, for instance through the development of SpaceFibre or the adoption of Ethernet-based ground technologies. The complexity of the resulting networks and protocols calls for appropriate measures for management. This paper presents the development of an upgraded version of the SPACEMAN network management tool, originally supporting SpaceWire Networks composed of NDCP-aware devices and non-NDCP aware SpW-10X routing switches, which was then extended to handle SpaceFibre networks and more models of non-NDCP aware devices. Directions of its further development for Time-Sensitive Networking are also outlined.**

Keywords—network management, NDCP, SpaceFibre, SpaceWire, RMAP

I. Introduction

In response to increasingly ambitious goals for future space missions and the associated needs of higher on-board data communication rates and quality, initiatives are on-going to upgrade the current communication technologies and add new services, for instance through the development of SpaceFibre or the adoption of Ethernet-based ground technologies. The complexity of the resulting networks and protocols calls for appropriate measures for management.

For SpaceWire such a measure was introduced in the form of the Network Discovery and Configuration Protocol (SpW-NDCP) [1]. One of the first implementations of SpW-NDCP together with a network management application called SPACEMAN was delivered by ITTI in a project for ESA [2,3]. The SPACEMAN application originally supported SpaceWire networks composed of NDCP-aware devices and non-NDCP aware SpW-10X routing switches.

Evolution of the application continued so as to support the new protocols and new device models, as well as enable cooperation with other applications. The paper presents the results of that progress which found their place in the revised SPACEMAN 2 application or are being developed, within the frame of the ESA-funded project *Multi-Protocol On-Board Communications Network Manager* (*MultiSpaceman*).

The major development direction is the support of more network protocols, beginning with SpaceFibre. In order to achieve this, the original SpW-NDCP had to be extended into a new version, described in a companion paper [4], with SpaceWire support retained though modified. Compatibility across versions is maintained: SPACEMAN 2 can handle both versions of NDCP. Pure SpaceWire or SpaceFibre as well as mixed SpaceWire/SpaceFibre Networks are supported. The next target network technology is IEEE 802.1-based Time Sensitive Networking (TSN).

The paper describes the key aspects of the advances made to SPACEMAN. Section II discusses the management protocols, their basic features, differences, and mechanisms they need to implement in order to support a network management application. Section III presents the device data model approach adopted in SPACEMAN to cope with the growing number of different incompatible devices. The devices supported currently by SPACEMAN are listed in Section IV. Other new features introduced to SPACEMAN are presented in Section V. Section VI outlines the directions for SPACEMAN development towards TSN. Finally, Section VII gives conclusions.

II. Protocols

Among fundamental tasks of a network management application are:

(i) discovering network device presence and identity,

(ii) discovering network topology,

(iii) discovering network device state,

(iv) discovering network device configuration,

(v) setting network device configuration.

These tasks can be performed using active means (sending network packets and observing the results); some of them (except task (v)) in for some network technologies can also make use of passive means (listening to network traffic generated by the devices).

Key aspects of network device addressing are the presence or absence of a unique device address and receiver addresses in command packets and sender addresses in reply

This work has been funded by the European Space Agency under contract no. 4000132938/20/NL/AS.

**Corresponding author*

packets. The SpaceWire standard [5] does not define a notion of a permanent unique device. Instead, addressing depends on the placement of the sender and receiver devices in the network – the network path between them. A related characteristic is the absence of the sender address at the level of SpaceWire packets. As a consequence, a management protocol for SpaceWire needs to be given network topology information in order to be able to address its commands. It also cannot rely on passive listening to network traffic (apart from the fact that SpaceWire switches generally lack supporting facilities, like port mirroring). Moreover, due to the possibility of network loops, it needs some form of a unique device ID to recognize situations where the same device is accessed by different network paths.

The management protocol dedicated to SpaceWire: the SpW-NDCP, takes those circumstances into account. It also adds the notion of device ownership by a management application (or control device), which prevents overwriting device configurations by non-owners. A network manager application that employs the NDCP, performs the network discovery process, during which it learns the network topology and the paths needed for addressing, and assigns each network device a unique ID. This requires support in the network devices of the NDCP, and in particular of the following registers (NDCP fields) on each device:

(a) a writeable one for the device ID,

(b) a writeable one for device ownership information,

(c) a readable one providing the number of the return port, i.e. the number of the port of the device through which the most recent command is received and the reply is sent back,

as well as an access control mechanism for protecting overwriting the device configuration by a non-owner.

Not many SpaceWire devices support the NDCP yet; examples of those that do include STAR-Dundee devices of the project that introduced the NDCP [6], Cobham Gaisler GR718B routing switches [7], and SpaceWire NDCP-aware nodes developed by ITTI in the *MultiSpaceman* project. An alternative protocol that could be used for network management is the RMAP [8].

It should be noted, however, that the RMAP, although standardised as to frame format and command-reply semantics, does not define the layout of the memory accessed. Different device models can have different register sizes, numbers, addresses, addressing modes (e.g. byte vs. word), byte order (most-significant vs. least-significant first), RMAP command types implemented (e.g. single address vs. incrementing address types), and, naturally, the values hold – their units, ranges, reset values, and meaning.

Access control similar to that available in the NDCP is not possible with the RMAP. As a result, network management should be constrained to a single control device (management application). Functions of the registers (a)-(c) discussed above need to be implemented as RMAP registers. The roles of registers (a) and (b) may be assigned to a single one, if there is only one that can be freely written by the application. Support of the functionality of register (c) is crucial for feasibility of network discovery via the RMAP. There are RMAP-aware devices without that functionality. When such a device is encountered, the problem of ambiguity of the return path appears, which can be solved by trying all possible return ports of the device; this approach is, however, inefficient and of limited scalability.

All those considerations are also applicable to SpaceFibre. The NDCP version 2, which has recently been proposed [4] and which is supported by the current version of SPACEMAN, is applicable to SpaceWire, SpaceFibre, and mixed SpaceWire/SpaceFibre networks.

III. DEVICE DATA MODELS

Originally the SPACEMAN network management application supported the SpW-NDCP and, as an exception, the RMAP as implemented in the SpW-10X switch. Since then, support for more device models has been added and the exception approach turned to a more systematic framework of device data models. The device data model specifies completely the method to access the device, including the protocol (SpW-NDCP, NDCP version 2, or RMAP) and, in the case of the RMAP, the availability and layout of the registers, the addressing mode, the destination logical address, and the RMAP key. The models to be used in a particular network discovery process form a prioritized list, which the user can set up by removing or adding models and changing their order (and thus priority).

There are four modes of selection of device data models by SPACEMAN:

- fully automatic: SPACEMAN tries each of the device data models on the list, until the device responds to an initial management command. The successful model is then adopted for further commands issued to the device. An auxiliary mechanism of a device signature can be used. The signature is the contents of specific bits of specific registers of the device that is considered a constant characteristic feature of the particular device model. This device data model is then used only if the appropriate device signature is found in the response to initial RMAP read commands (issued according to the currently tested model); otherwise the next model from the list is tried. If there is no reply from the device after trying all the models from the list, and the link to it is active, the device is marked ‘generic’: no information on it is available except its existence and place in the network.
- semi-automatic: This is similar to the fully automatic mode. However, if there is no reply from the device on an active link for any of the models on the list, the user can manually select a model to use for this device (supposedly a model that was not initially included in the list of models to be used).
- manual: For each device, before trying to send an RMAP query and get any reply, SPACEMAN asks the user to select a model to use for the query.
- manual in every cycle (relevant only in continuous discovery operation): This is similar to the semi-automatic mode. While in the (plain) manual mode SPACEMAN remembers the user's selection of the model for a specific device, in this mode such a selection needs to be made every time any device is encountered (even if a model for it was previously selected).

IV. Device models supported

There are two aspects of device support by SPACEMAN. A device model can be supported as part of the managed network. Supporting this role of the device requires implementing communication between SPACEMAN and the device using any of the management protocols (NDCP or RMAP).

A device is also needed for SPACEMAN to physically connect its host computer to the managed network. This can be an internal SpaceWire board (e.g. PCI-based) or an external device with a SpaceWire interface and an interface connected directly to the host, like USB or Ethernet. Such a device is called the management gateway device; SPACEMAN communicates with it using an API or some dedicated protocol provided or documented by the device vendor. The range of supported management gateway device models was augmented, and they can now be monitored by SPACEMAN, with continuous display of the connectivity status of their network ports.

Table I lists the device models currently supported by SPACEMAN. For SpaceFibre devices, features specific for this technology, like virtual channels/networks and lanes are supported. In addition to real devices, there is also support under development for devices simulated in MOST-X.

Table I. Device support by SPACEMAN

Device	**Discoverability and manageability as a network peripheral device?**		**Capability of a management gateway device?**
	NDCP	RMAP	
STAR-Dundee SpW NDCP-aware devices of the original NDCP project	yes	yes	yes
STAR-Dundee SpW and SpFi devices with STAR System API 3.10 or 4.0	via a USB-or PCI/PCIe-connected PC-based emulator	yes	yes
STAR-Dundee SpW-USB Brick Mk1	no	yes	no
STAR-Dundee SpFi and SpW Router Breadboard	no	yes	no
SpW-10X switch	no	yes	no
TELETEL iSAFT SpW and SpFi PCIe boards	no	no	yes
Shimafuji GPWGB0012 switch	no	yes	yes
ITTI SpW node	yes	yes	yes
ITTI SpFi node	yes	no	no
Cobham Gaisler GR718B switch	in development	in development	no
Thales Alenia Space MOST-X simulator	in development	yes	in development

V. Other new features

Other features developed recently include facilities for cooperation with other applications as well as various user interface additions and improvements.

SPACEMAN now includes a TCP/IP server that can exchange device models (currently in the XML format) with external applications. The models can be sent both ways: SPACEMAN can send a model it has in its memory, whether just discovered from the connected physical network, created by the user in the built-in editor, or read-in from a file; SPACEMAN can also receive a model from an external application and do with it whatever could be done with a model read from file, e.g. use for comparison against another network (possibly the one being just discovered).

There are a number of user interface additions. While SPACEMAN sends packets to a network or receives them, it can be paused by a breakpoint, just before or just after sending or receiving. The user can then inspect the packet in the log and optionally edit its contents before resuming the paused transmission. Selected or all packets in the on-line packet log of SPACEMAN can be exported to a file.

Operations involving packet transmission, like network discovery, automatic configuration, or sending/receiving individual packets are equipped with time-related facilities. There are various options for the start time, end time, and repetition of an operation.

Finally, a SpaceWire/SpaceFibre network can be accessed and managed remotely, with the SPACEMAN application operating locally on the user's computer and a remote auxiliary connector application interfaced directly to the managed network while communicating with SPACEMAN over a TCP/IP link.

Fig. 1 displays a screenshot of SPACEMAN, showing the status bar of two management gateway devices, the tree and diagram of the discovered network, the log window, and the status bar with breakpoint controls and packet counters.

VI. Beyond SpaceWire/SpaceFibre

Following an increasing interest in application of TSN to on-board data networks, a follow-up activity is being prepared to extend SPACEMAN's applicability domain to TSN. This is a major development, since some properties of the Ethernet networks that are essential for network management are quite different than those described in Section II. The devices have a permanent unique MAC addresses, the data frames carry them as sender and receiver addresses (rather than paths), there may be a possibility of passive listening to traffic such as Link Layer Discovery Protocol. The management protocols are naturally quite different; Simple Network Management Protocol, NETCONF, or RESTCONF may be used for different devices to read and write configuration parameters. Moreover, because of the focus on time sensitivity and deterministic transmission, there are dedicated protocols for data stream definition and scheduling, like the IEEE 802.1Qcc [9], 802.1Qbv [10], and more of the 802.1 standard group. This will add network traffic design tasks to SPACEMAN with, performed either in the application or in cooperation with external applications. Fig. 2 shows SPACEMAN as part of the system with a network, external applications, connections, and protocols.

VII. Conclusion

Recent developments in the SPACEMAN network management application were presented, together with future work directions. It should be noted that network management is strongly dependent on the details of protocol support in the devices managed. For a network management tool, handling multiple protocols is inevitable, taking into account the variety of devices in use. It is worth noting, that tiny details in those devices, like the absence of the return port register in an RMAP implementation, can significantly affect the tool's operation and efficiency.

REFERENCES

[1] European Cooperation for Space Standardization, Space Engineering – SpaceWire Network Discovery & Configuration Protocol, ECSS-E-ST-50-54 Draft 1.8. Noordwijk: ECSS Secretariat, 2016.

[2] W. Hołubowicz, P. Lancmański, K. Romanowski, V. D. Kollias, and N. Pogkas, "SPACEMAN: A SpaceWire Network Management Tool," Proc. 6th Int. SpaceWire Conf. Athens 2014, pp. 99-102.

[3] K. Romanowski, P. Tyczka, W. Hołubowicz, R. Renk, V. D. Kollias, N. Pogkas, and D. Jameux, "SpaceWire Network Management Using Network Discovery and Configuration Protocol," Proc. 7th Int. SpaceWire Conf. Yokohama 2016, pp. 45-50.

[4] J. Kwiatkowski, K. Romanowski, P. Tyczka, and D. Jameux, "SpaceFibre and SpaceWire network management: NDCP version 2," submitted to 9th Int. SpaceWire and SpaceFibre Conf. Pisa 2022.

[5] European Cooperation for Space Standardization, Space Engineering – SpaceWire – Links, nodes, routers and networks, ECSS-E-ST-50-12C Rev.1. Noordwijk: ECSS Secretariat, 2019.

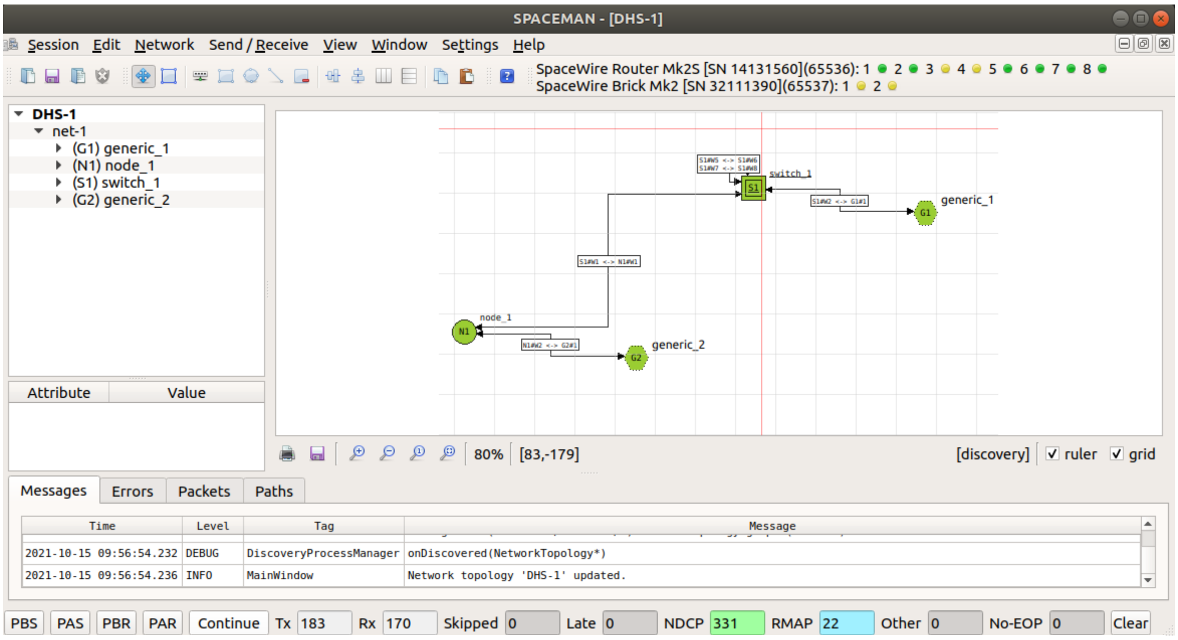
[6] S. Fowell, "Network discovery protocols – final presentation", presentation at the TEC-ED and TEC-SW Final Presentation Days, Noordwijk, 2014.

[7] GR718B Radiation-Tolerant 18x SpaceWire Router. 2020 Data Sheet and User's Manual, GR718B-DS-UM version 3.5, Cobham Gaisler, 2020.

[8] European Cooperation for Space Standardization, Space Engineering – SpaceWire – Remote memory access protocol, ECSS-E-ST-50-52C. Noordwijk: ECSS Secretariat, 2010.

[9] Institute of Electrical and Electronics Engineers, IEEE Standard for Local and Metropolitan Area Networks – Bridges and Bridged Networks – Amendment 31: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements, IEEE Std 802.1Qcc-2018.

[10] Institute of Electrical and Electronics Engineers, *IEEE Standard for Local and Metropolitan Area Networks – Bridges and Bridged Networks*, IEEE Std 802.1Q-2018.



SPACEMAN - [DHS-1]

SpaceWire Router Mk2S [SN 14131560](65536): 1 ● 2 ● 3 ● 4 ● 5 ● 6 ● 7 ● 8 ●
SpaceWire Brick Mk2 [SN 32111390](65537): 1 ● 2 ●

- DHS-1
  - net-1
    - (G1) generic_1
    - (N1) node_1
    - (S1) switch_1
    - (G2) generic_2

80% [83,-179] [discovery] ☑ ruler ☑ grid

```
Time                    Level  Tag                      Message
2021-10-15 09:56:54.232 DEBUG  DiscoveryProcessManager  onDiscovered(NetworkTopology*)
2021-10-15 09:56:54.236 INFO   MainWindow               Network topology 'DHS-1' updated.
```

Tx 183 Rx 170 Skipped 0 Late 0 NDCP 331 RMAP 22 Other 0 No-EOP 0

Fig. 1. Example screenshot of the SPACEMAN application during work

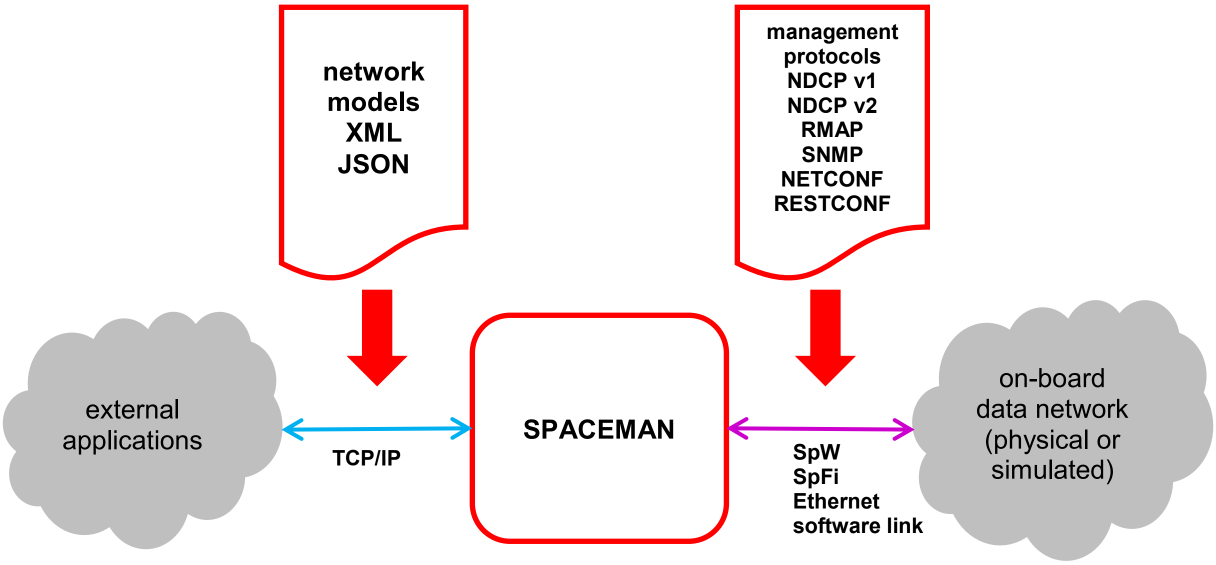


Fig. 2. SPACEMAN in the context of managed networks, external applications, and protocols

Tuesday 18th October

Networks & Protocols 1 (Long)

Deterministic SpaceWire / SpaceFibre ?

The Challenge !

Fotis Kostopoulos
TELETEL SA
Athens, Greece
f.kostopoulos@teletel.eu

Nikos Pogkas
TELETEL SA
Athens, Greece
n.pogkas@teletel.eu

Vangelis Kollias
TELETEL SA
Athens, Greece
v.kollias@teletel.eu

Barthelemy Attanasio
THALES ALENIA SPACE France
Cannes, France
barthelemy.attanasio@thalesaleniaspace.com

Hans-Joerg Beestermoeller
AIRBUS Germany
Bremen, Germany
hans-joerg.beestermoeller@airbus.com

Hans Corin
BEYOND GRAVITY
Gothenburg Sweden
Hans.Corin@beyondgravity.com

Felix Siegle
European Space Agency (ESA)
Noordwijk, Netherlands
felix.siegle@esa.int

***Abstract*— This paper presents the results of an activity led by TELETEL in the area of onboard data networks aiming at defining a deterministic protocol layer for SpaceWire and SpaceFibre networks. The main objective of the activity is the development of a novel deterministic protocol layer for SpaceWire and SpaceFibre that works primarily at network level (i.e. within routing switch devices), allowing therefore to build fully deterministic networks with both protocol-aware and legacy SpaceWire/SpaceFibre nodes. The work is performed under the ESA study entitled "SpaceWire Network Management Service Suite definition and validation".**

Keywords—Spacewire, SpaceFibre, SpW, SpFi, TTE, TSN, AFDX, CBS, BAT

I. Introduction

The new concept elaborated by ESA for spacecraft avionics systems is bundled under the common Space Avionics Open Interface Architecture (SAVOIR). SAVOIR working group has issued a requirements document addressing the on-board communication system [1] where the term 'Deterministic / Determinism' is clearly defined. The objective of the activity presented in this paper is to introduce a novel deterministic protocol layer to the SpaceWire and SpaceFibre on-board networks according to the SAVOIR guidelines. However this results not only in functions ensuring the transmission within a certain time-window (SAVOIR definition), but also in the need for FDIR, network management, quality of service, etc., which are derived from system requirements (e.g. closed-loop control for AOCS). The applied control algorithms need to take into account the uncertain latencies that are a result of a non-deterministic network. For example, the time a sensor value was taken, until its computation inside the loop and the latencies to transmit the control command to the actuators. When drawing the chain of contributors, it is obvious, that the communication system is an essential element.

The MIL-STD-1553 protocol was and it is still used to provide determinism but for low data rate networks. Network topology has also evolved from the single master, as for MIL-1553 to multi master and switched networks. TTEthernet, SpaceWire and SpaceFibre are systems used in modern on-board architectures providing a much higher net data-rate. TTE is baselined for Ariane 6 and in use for the European Service Module for ORION. European contributions to the Lunar Gateway have baselined TTE such as IHAB in the respective systems as well. While combining high-speed communication, up to 1Gbit/s, with determinism, TTE appears to be very suitable. However, the effort for planning and configuring the network as well as for the verification process turned out to be high.

SpaceWire (and in the future SpaceFibre with much higher performance) is widely used in on-board architectures. SpaceFibre, in comparison with SpaceWire, is bringing already the possibility of a significantly higher data-throughput combined with means regarding the quality of service. However, determinism, as defined by [1], is not implemented yet in native SpaceWire. In order to extend the capabilities of SpW, some ideas have already been pursued resulting in the definition of N-MaSS for adding FDIR, a certain determinism with SpW-D or SpW-R targeting reliability. These concepts are adding an additional protocol layer (SpW-R, Spw-D) or additional monitoring functions (N-MaSS) to enable the functions.

II. Initial requirements for a deterministic SpaceWire/SpaceFibre protocol

According to ESA [2], the main means of realizing determinism shall be to add an additional network layer as shown in the figure below.

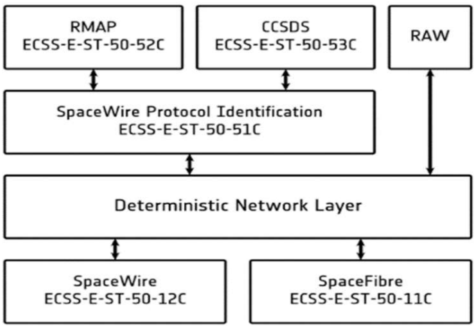


Fig. 1. New protocol layer in the existing SpW/SpFi protocol stack

Any feasible design shall be capable of handling new deterministic and other legacy nodes, requiring thus an awareness of the protocol in the switches and the end nodes. The new protocol layer shall be developed in a way that it is mainly managed by the routing switches to keep compatibility with legacy nodes. New deterministic nodes may manage the new protocol layer directly while the new deterministic switches shall act as translator for legacy nodes. Traffic from and to legacy nodes shall be extended or stripped by the switch from the new deterministic layer. That way the switches act as translator between new and legacy protocols if necessary.

Deterministic and time accurate messaging requires all devices in the network to work with the same time basis. Since the native time synchronization mechanism on raw SpaceWire is not robust against failures such as delayed or erroneous time codes, a new concept shall be developed. The new time-code mechanism shall allow more robust and accurate time-code distribution in SpaceWire and SpaceFibre networks.

For deterministic network communication reliability is important. Therefore, the new deterministic layer shall also implement a concept for Fault Detection, Isolation and Recovery (FDIR). This shall include controllability of the new network layer from the user application as well as through the network. Optionally with a basic watchdog mechanism to control the status of the user applications through the network.

Before deriving a feasible approach for a network layer, a comparison with other approaches has been performed. For this activity and further to the past SpW initiatives (SpW-D, SpW-R, N-Mass) the approaches used in systems like AFDX, TTE and TSN have also been taken into account.

III. Comparison between protocols

From the analysis performed with the different approaches around determinism in the frame of other protocols than SpaceWire, it can be obviously concluded that depending on the need, the solution of mechanisms is not the same.

In fact, for example, AFDX does not bring a high level of determinism but at least controls the flow and ensures a bandwidth for every emitter. On the other hand, the 1553 defines every slot to communicate for every endpoint so it is fully deterministic by design but cannot bring any flexibility in flight due to its definition of traffic in a cyclic way.

Then, the two most interesting protocols which ensure a high level of determinism and flexibility in a network are the TSN and TTE which are both based on Ethernet. They are based on accurate synchronization of all the equipment within the network. The mechanisms implemented and defined in several norms ensure determinism. The interesting way to develop deterministic for SpaceWire is to take the advantages of these mechanisms without bringing the drawbacks such as the accurate synchronization, which adds several constraints to the implementation.

Comparing both protocols and mechanisms, after some research on previous studies and on research papers, it shall be noticed that there are many differences in terms of performance even if both are based on Ethernet. In fact, focusing on what is important in terms of performance but leaving the industrial problems apart, TTE seems to be a rather closed standard with only 3 priorities (TT, RC, BE) and without any flexibility.

One of the most important and famous mechanisms of TSN is the Credit-Based-Shaper (CBS), which allows the transmission of a frame when a credit is available for the associated class. The paper [3] concludes on some interesting performances of the CBS such that it can bring flexibility into the traffic because if a class does not use its bandwidth allocated, a lower priority class can use it. Its role is to average the communication delays, in fact, comparing to other time-triggered protocol, the latency of high priority traffic is higher but it ensures that low priority traffic has at least a communication window to send their packets. To improve the communication latency, TSN defined a new traffic shaping mechanism to accommodate strict real-time transmission with deterministic end-to-end delays, which was the Time Aware Shaper (TAS), but this required a high accurate synchronization of all the nodes. Network Calculus theory has been performed in such shaper to be able to compute the worst-case latency between two nodes on a dedicated topology.

IV. Final Requirements Consolidation

A. SpW Working Group Questionnaire Highlights

- One of the first and most important thing is that any development shall ensure the **compatibility with legacy SpaceWire nodes**. This is something mandatory. In fact, the development shall not modify any lower SpaceWire layers neither modify the corresponding ECSS.
- Regarding the topology and the roles of the endpoints, it is clear that a SpaceWire network needs to have **multiple masters and multiple slaves**.
- Moreover, one usual topic is the question of including the FDIR in the additional layer which has a more balanced result. In fact, **two-third were in favor of having the FDIR managed by this new layer**.
- Multicast SpaceWire: the problem is that if one of the ports is blocking, all the ports are out of service. GAR (group adaptative routing) is considered “useful”.
- The **packet size** has been decided to be adjustable and not fixed by the protocol.
- The **case of arbitration at switch level** could be kept with the Round-Robin or adding a new mechanism such as a priority.
- The majority (two-third) chose for **increasing the buffer size** in SpW switches as it is often the limitation of the network creating congestion. The question is why, because there are already FCTs. Even with FCTs, the low size can generate congestion if there are multiple high data rate links.
- What about the **Store & forward** in SpW? 50% were pros and 50% were cons. Store & forward + broadcast seems mandatory in high-level protocols based currently on Ethernet.

B. Requirements extraction from ESA SoW, SpW-NMaSS, SpW-D

Requirement	
Determinism	The new deterministic network protocol shall ensure data transmission on the SpW or SpFi network within a certain time frame with upper and lower bounds.
Quality of Service	The network protocol shall provide several quality of services, such as (priority QoS, best effort QoS, guaranteed QoS, scheduling QoS)
Network layers	An additional network layer on top of the SpW and SpF protocol shall enable the determinism of the deterministic network.
Legacy	Legacy SpW or SpF end nodes shall remain operable in the deterministic network.
Common time basis	The network shall provide a new time-code mechanism that provides all devices in the network with the same time basis, which is accurate and robust against failures such as, delayed or erroneous time codes.
FDIR	The new network layer shall implement a concept for Fault Detection, Isolation and Recovery (FDIR).
N-MaSS Concept	The network FDIR shall be based on the N-MaSS concept (network management servicing suite for FDIR).
FDIR message types	The network FDIR shall send and receive messages in order to monitor the network status. The messages shall consist of the following: • Health status request and response messages • Combined messages of data and health status • Switch configuration messages (read/write)
Implementation of FDIR handler	The network FDIR handler is implemented in switches and nodes.
FDIR actions	FDIR actions are based on specified error signatures and are pre-computed.

C. Requirements from ADS missions and use cases

Requirement	
Bandwidth	The SpW network shall provide a bandwidth of 400 Mbit/s
Latency	It shall be possible to have latencies of data packages below 1ms (TBC)
Data acknowledgement	Reception of all data packages shall be acknowledged.
Time tagging	All data packages shall be time tagged based on a global temporal basis.
Loop Through Device	It shall be possible to establish a loop through device in order to enable the implementation of bus-like architectures with sequential topology of network nodes.
QoS	It shall be possible to provide different QoS.
Failure protection	The data transmission shall be protected against failures depending on the selected QoS.

D. Requirements from SAVOIR and TAS missions

Requirement	
Guaranteed message transmission	Guaranteed message transmission shall be assured for certain message types (QoS)
Time tagging	Time tagging of messages shall be foreseen.
Limited message delay	The maximum message delay shall be limited.
Time tag accuracy knowledge	The accuracy of the time tags shall be known.
SpW Timecodes	SpaceWire timecodes shall have higher priority than any other message type.
Time code jitter	Time code jitter shall be bounded by TBD ms
Time code latency	Time code latency shall be not larger than TBD ms.
Selectable bandwidth	It shall be possible to allocate different bandwidth between different links.
Bandwidth	A bandwidth of 200 Mbit/s shall be reached.
Prioritisation	It shall be possible to prioritize messages in flight without interruption or reconfiguration.

E. Main driving requirements synthesis

To conclude on all the requirements that have been discussed in the previous sections gathering how the determinism is accessible in some other protocols and what is needed in the different use cases, it is possible to select basic notions and features, which are necessary to reach a full deterministic SpaceWire protocol as requested in the specific use cases. Among these features, ADS and TAS converged on most of the same features, choosing the following ones:

- Guaranteed bandwidth – **mandatory**. It enables to be sure that a minimum bandwidth is available for one end system to communicate at least some data.
- Guaranteed delays – **mandatory**. It ensures that a message will be received not later than a certain time. This is useful for real-time computing.
- Priority management – **optional**
- High resolution time synchronisation service – **mandatory**. This can be used to keep the scheduler of multiple remote Time Partitioning Systems in sync. In fact, precise synchronisation allows to move applications/partitions from one system to another without losing the predictability of the system.
- Acknowledgement on data transfer – **mandatory for a full determinism**. This ensures to have a confirmation of the reception of the message. It is something important, if not the most important thing to ensure a full determinism.
- A deterministic traffic class that guarantees the delivery within a defined time slot (for a full determinism for at least one kind of data, this could be an alternative to the previous point). Since all partitions on the TPS have allowed time slots during which they are allowed to run packet delivery needs to happen with deterministic timing. That way the needed data is available during the active time of the partition.

Other features are important and would be interesting to implement such as

- Session management (health check & FDIR)
- Compatibility with HW or SW-based end-users
- Bandwidth efficiency
- Multi-initiators

- Centralised management
- Protection against failure

In any case, it has to be noted that the ability for legacy devices to send non-deterministic traffic through the network without interference with the deterministic traffic is necessary for a full compatibility with ancillary HW.

F. *Comparison of existing SpW protocols*

This section gathers the main SpaceWire protocol layers, which try to provide a certain degree of determinism over the native SpaceWire, which includes the Physical, Data Link and Network Layers. Each protocol has been analysed but are now assessed on different topics. Comparing to the different features required for having the ideal deterministic SpaceWire, the following table presents the comparison between each protocol.

	Native SpW	SpW-R	SpW-D	SpW-NMaSS	STP-ISS
Guaranteed bandwidth - mandatory	No	No	Yes	No	Yes
Guaranteed delays - mandatory	No	No	Yes	No	Yes
Priority management – optional	No	No	Yes	No	Yes
Acknowledgement on data transfer – mandatory	Enabled (RMAP)	Yes	Yes	?	Yes
Session management (health check & FDIR)	No	No	No	Yes	Yes
Compatibility with HW or SW-based end-users	Yes	Yes	HW only	?	Yes
Bandwidth efficiency	Yes	Yes	No (no TS overlap)	?	Yes
Multi-master management	Yes	Yes	Yes	Yes	Yes
Multi-initiators	Yes	Yes	Yes	Yes	Yes
Centralised management	No	No	No	Yes	Yes
HW-SW complexity	Low	?	Imposes RMAP & HW end-users for quick replies within TS or multi-polling per transaction	Requires specific HW & SW	Requires specific HW & SW
Protection against failures	No	No	No	?	No
Compatibility with anciliary HW	Yes (reference)	Yes	Yes	Yes	No

V. Proposed Protocol Architecture

The overall layers used in the different SpaceWire networks can be summarized as presented in the following diagram where the SpW IP block gathers all the SpaceWire standards. At the left the switch IP and on the centre and at the right the SpW IP of an End Node. Regarding the switch, the proposed SpW NMS(in light green) can replace the native Wormhole Routing and is a new behaviour proposed within this study.

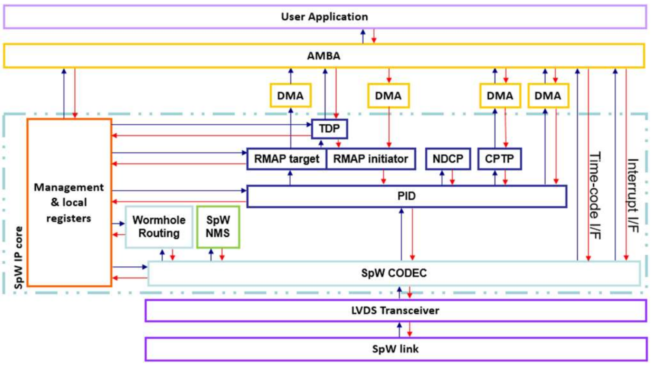


Fig. 2. SpaceWire layers with new proposed deterministic layer

The above layers represent the interaction with a SW through the AMBA bus to connect processing subsystem to programmable logic. The bottom layers refers to physical layers of the SpaceWire standard. Then, all the layers inside the dash light blue square belong to SpW. The dark blue blocks are optional standards that can be used alone or together.

Therefore, instead of having a wormhole routing in a topology with SpaceWire, this solution introduces a store and forward routing switch with ingress policing, packet filtering and traffic shaping to support fault tolerance and determinism in the network. The implementation of the new SpW deterministic routing switch is more complex than a simple wormhole switch. In fact, the features and configurations are numerous and has to be well handled. This can be summarized in the following figure where two End Nodes exchange CPTP SpW packets.

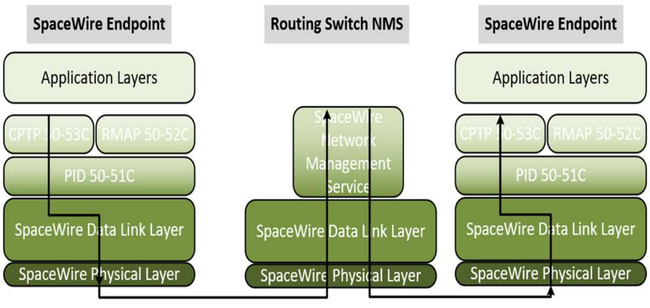


Fig. 3. SpaceWire stack in a communication through a routing switch

As expected from the analysis and requirement document [4], the SpW CODEC is not modified in order to enable backward compatibility with all the existing units using SpaceWire. In addition, new features are required to enable the following capabilities:

- Stream Identification and priority assignment
- Packet Size protection

- Store and forward
- Priority queues at output ports
- Traffic shapers

A high level diagram of a switch implementation forwarding packets from input ports to output ports is presented in the following figure.

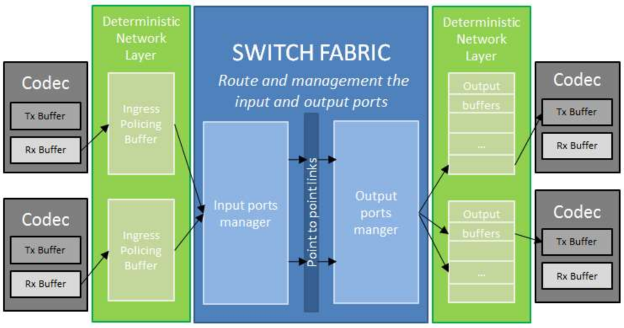


Fig. 4. Simple Schematic of a Routing Switch Implementation

A brief description of the proposed services and functions are presented below.

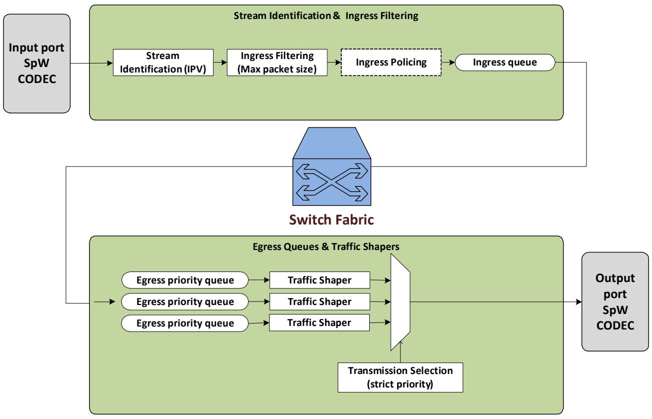


Fig. 5. Overview of switch services and functions

Stream Identification: Assign an IPV to the packet based on input port or based on packet fields.

Ingress Filtering: Drop or Truncate with EEP the packets with size larger than the max length allowed for this input port or for this IPV.

Ingress Policing: Ensure ingress flows meet their specifications, mark frames that are out-of-spec and drop, count for diagnostics, etc. (future extension).

Ingress Queue:

- One ingress queue at each input port
- Temporarily store input packet pending reception (EoP) or input packet(s) pending switching
- Capacity at least one packet of maximum size with any additional space to compensate technological delays

Switch Fabric:

- Decide set of egress queues based on packet's destination address and IPV
- Transfer the packet from the ingress queue to the destination egress queue if not full
- If egress queue is full and GAR is used, select an alternative egress queue
- If egress queue is full drop the packet or pause the reception from the input port
- If more than one ingress queues have a packet to be transferred at the same egress queue select the packet based on a first-in-first-out arbitration or based on round-robin arbitration

Egress Priority Queues:

- At least three queues at each output port
- One queue per used priority IPV
- Temporarily store packets pending transmission
- Required capacity depends on scenario and has a dependency on shapers configuration and traffic profiles

Traffic Shaper:

- Allow or deny transmission of packets from this egress queue based on bandwidth and timing criteria
- Two shapers are proposed CBS or Bandwidth Allocation Table (BAT)
- Other shapers can be defined / used in future

Transmission Selection:

- Select the packet for transmission at this output port from the egress queues based on strict priority criteria

A. Processing at the switch

The use of the store and forward switching means that the packets are fully received by the Switch and are then checked for consistency, packet size, etc., until they are placed into the dedicated output queue. This will impose a reception delay at the switch that depends on the packet size and link speed.

As an alternative cut through switching is also proposed to reduce forwarding latency in cases when the output queue is empty, where the switch immediately forwards the packet if the header (path/logical address) is received. This mechanism is commonly used in Ethernet switches but is not considered in the current simulation and demonstration activities due to implementation complexity.

A conflict (whatever nature) can cause an additional delay at the switch until the transmission resource (port/link) becomes free. In such case, the pending packets are buffered, and their transmission is deferred. A packet delay has different impacts on different traffic priorities.

- Best-Effort (BE): does not have any timing guarantees
- Among different priorities the priority mechanisms resolve the conflict. The high priority traffic will be selected over a lower priority.

- Among the same priority some latency / jitter will be introduced.

VI. Prototype Implementation

In the scope of the SpW NMS project TELETEL developed/implemented the proposed deterministic SpW Switch. The IP blocks complies with the candidate protocol requirements as defined in deliverable D1 (Requirements Consolidation Report), section VIII [4].

The block diagram of the deterministic switch is presented in the following figure. The Switch comprises of 8 SpW ports (at maximum) all connected to the Router's switch fabric. The SpW NMS Router is a store and forward router responsible for routing SpW packets to the appropriate output port. The routing is performed according to the SpW standard (via Physical/Logical address processing), accompanied by a priority stream policy, which indicates the priority level of each packet. In addition, the Ingress and Egress blocks at the input and output of the ports (respectively), implement the logic that enables deterministic behaviour.

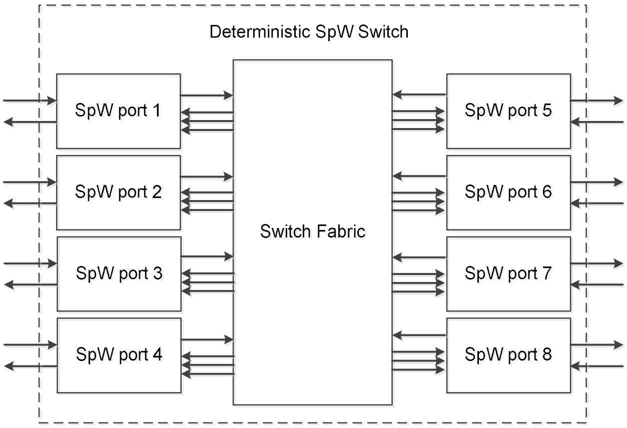


Fig. 6. Deterministic SpW Switch block diagram

The next figure presents the architecture of the **SpW port block**. This block implements a SpW CODEC that handles the interface with the SpW link (along with all related logic i.e. the necessary transmission/reception logic), as well as the Ingress and Egress blocks which are responsible for the deterministic behavior of the Switch.

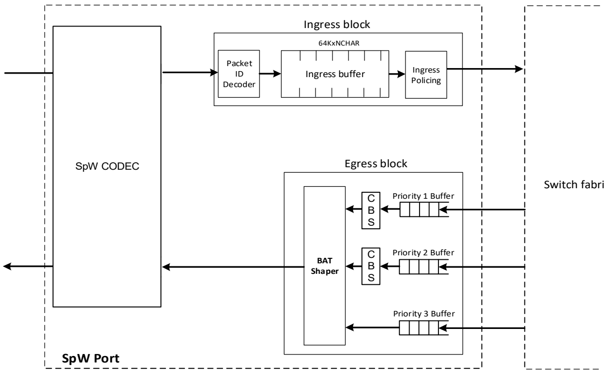


Fig. 7. SpW port block diagram

Each SpW port block consists of the following sub-blocks:

SpW CODEC (implemented by TELETEL SA and already used and validated in all company's SpaceWire products and in several space missions), which implements the ECSS-E-ST-50-12C SpaceWire standard.

Ingress Block, which performs stream identification and packet size policy and consists of the following sub-blocks:

- <u>Packet ID Decoder:</u> This logic checks the ID field (or port priority) and marks the packet with the corresponding priority, so that it will be written in the correct queue on the egress block. The priority is stored before routing and is used to generate the request to the appropriate Egress queue.
- <u>Ingress Buffer</u>: The Ingress Buffer is responsible for storing incoming packets. The ingress buffer can store at least one packet of max size
- <u>Ingress policing:</u> This logic checks the size of the packet being received. If the size exceeds the configured max packet size, the packet is either:
    - Dropped, i.e. the Ingress buffer is flushed
    - Truncated at max packet size, terminated with EEP character, and the rest of the packet's bytes are discarded, until an EoP is detected.

NOTE: This option is configurable by register.

Egress Block, which implements the priority queues and traffic shaping and consists of the following sub-blocks**:**

- <u>Egress Buffers</u>: The Egress Buffers are responsible for storing packets prior to transmission. 3 Egress buffers are implemented, one for each priority queue.
- <u>Credit based shapers - CBS:</u> Credit based shaping is implemented. The shaper keeps track of the sending and idle slopes and grants access for transmission to the appropriate queue. Priority queue 3 does not have a CBS as it implements BE traffic.
- <u>Bandwidth Allocation Table - BAT Shaper:</u> The BAT Shaper selects the priority queue that will transmit next according to a configurable bandwidth allocation table that specifies which queue can transmit at specific time slots. If the BAT Shaper is disabled, a fixed priority arbiter with a strict priority is used (i.e. priority queue 1 is always selected prior to the other queues).

The next figure presents the architecture of the Switch fabric block. This block is responsible for routing the incoming packets to the appropriate priority queue of the output port.

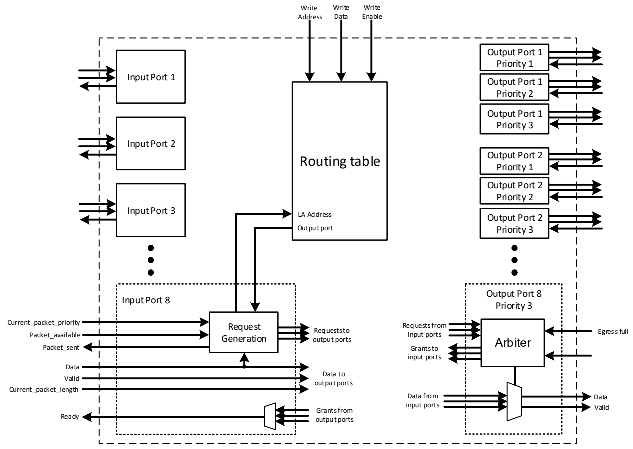


Fig. 8. Switch fabrix block diagram

This block consists of the following sub-blocks:

Input port block, which is connected via a FIFO interface with Ready/Valid handshake policy to the Ingress block. It comprises of the following sub-block:

- Request generation: The combination of Routing table information (output port request according to Physical/Logical address) and the packet priority value are used in order to generate the request to the appropriate priority queue of the output port.

Routing table: The Routing table is a local memory that indicates the output port that the packet should be routed according to the Physical/Logical address. The table is configured by the SW application before routing any SpW packet. The Request generation logic uses the packet's address to search for the output port and generates the request. In case two or more input ports require access to the Routing table simultaneously, a round-robin based arbitration is performed.

Output port block, which is connected via a FIFO interface with Ready/Valid handshake policy to the Egress block. This block is connected to a priority queue of an output port. It comprises of the following sub-block.

Arbiter: The arbiter receives requests from all input ports and if simultaneous requests are received by two or more input ports (i.e. two ports request access to the same output port priority queue simultaneously), the arbiter will grant access to one of them according to Round Robin arbitration, until the entire packet is routed to the output buffer (wormhole implementation). The arbiter keeps track of the capacity of the priority queue and if the priority queue has free space (the entire packet can be stored in the queue), the arbiter grants access to the queue.

VII. Demonstration Plan

This section presents the preliminary demonstration scenarios to be performed in a real SpW network testbed using SpW Nodes (up to 8) and SpW Switches (up to 2) implementing the candidate protocol features to be performed in the scope of WP6 (Demonstration) by TELETEL.

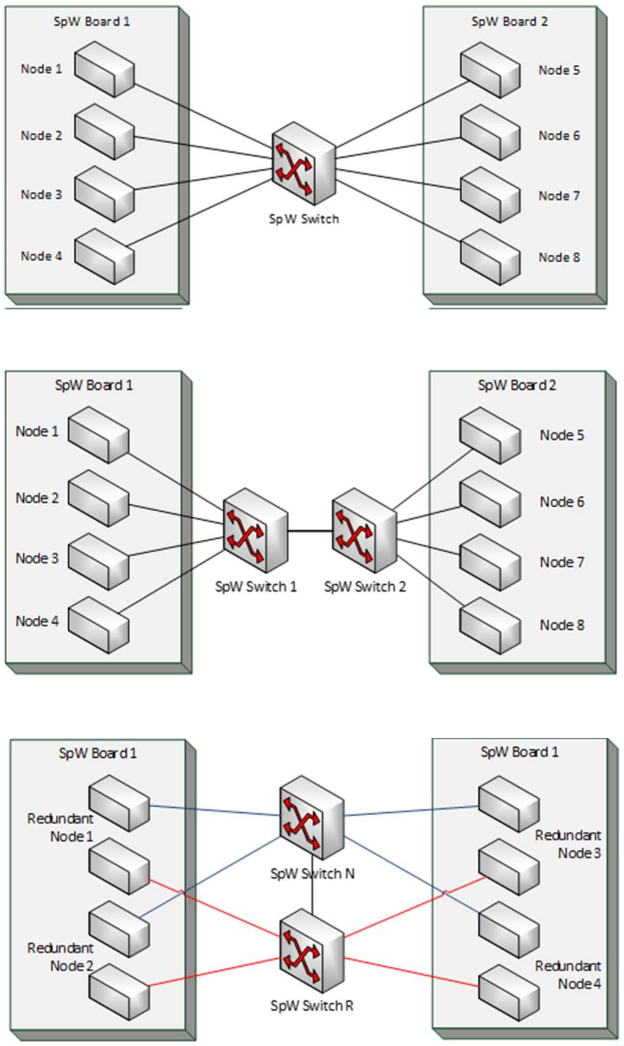


Fig. 9. Possible topologies for verification and demonstration scenarios

The following tests will be performed on the final demonstration of the SpW NMS network. The tests are divided in the following categories:

- Unit tests. These tests aim to verify the correct functionality of individual features / requirements of the SpW NMS router. Each test shall use a simple scenario to validate a single (or a few) functional features of the router. A topology with a single router shall be used.

- Network performance measurement tests. These tests shall be used to quantify the performance of the NMS router in a network. The performance characteristics of the network shall be validated (switching latency, end to end latency, packet jitter, bandwidth etc.). Topologies with a single router and two routers in series shall be used.

- Legacy device support tests. The tests shall validate the routers behaviour when legacy SpW devices are connected. A topology with a single router shall be used.

- Error cases tests. The tests shall validate the router's functionality in network error cases (babbling idiot nodes, congestion cases, disconnects etc.). A topology with a single router shall be used.

- Full demonstration scenarios. The tests shall validate the router's functionality and performance in full demonstration scenarios as defined in the network simulation activity (in order to directly compare the results between the simulation and the real implementation). Topologies with a single router and two routers in series shall be used.

The following functional parameters shall be used for all tests:

- The link rate for all ports shall be up to 100Mbps. This link rate is representative of most SpW application scenarios and is enough to demonstrate all scenarios. Packet to packet delay shall be used to limit individual port traffic bandwidth. For selected test cases, scenarios with link rates of 10Mbps will also be executed (to validate functionality/performance with low link rates)
- The max packet size for all ports shall be set to 32KB or 16KB. A mixed approach is proposed (1 or 2 ports set at 32KB, the remaining ports set to 16KB or lower). The final configuration (or multiple configurations) of the router shall be determined by the FPGA resources when implementation is complete and the test steps are defined.

VIII. Conclusions

In this paper, we present the work that has been carried out in the ESA study "SPACEWIRE NETWORK MANAGEMENT SERVICE SUITE DEFINITION AND VALIDATION". The aim of the study was to define a deterministic network layer for SpaceWire and SpaceFibre networks by combining different approaches that have been used in other deterministic networks such as TTE, TSN with previous attempts to add determinism to SpaceWire (SpW-D, SpW-R, N-Mass). To achieve this, requirements from current and future missions have been considered, mainly from TAS, AIRBUS and BEYONDGRAVITY (former RUAG) experiences.

The proposed deterministic network layer has been already simulated with different scenarios in the MOST simulator by TAS. Prototype implementation in FPGA has been carried out by TELETEL and the resulting deterministic SpW router has been integrated in the target evaluation boards by TELETEL (TELETEL S.A. – Octal SpaceWire PCIe Interface Card).

The activity started in March 2020 and is expected to be completed in November 2022. Several demo scenarios will be executed at an experimental testbed at TELETEL premises aiming at assessing the correctness of the approach. The results of the study will be available to the SpaceWire/SpaceFibre community in December 2022.

Acknowledgment

The described work has been funded by the TDE program of the European Space Agency (ESA).

References

[1] SAVOIR On-board Communication System Requirement Document (SAVOIR-GS-008)

[2] SPACEWIRE NETWORK MANAGEMENT SERVICE SUITE DEFINITION AND VALIDATION - EXPRO+ (SOW)

[3] Feng He, Lin Zhao, Ershuai Li, "Impact Analysis of Flow Shaping in Ethernet-AVB/TSN and AFDX from Network Calculus and Simulation Perspective"

[4] SpaceWire NMS study, D1: Requirements Consolidation Report, User Requirements Document, Protocol Architectures & Trade-Offs, Issue 2

[5] SpaceWire NMS study, D3: Protocol Architecture Definition Report

[6] SpaceWire NMS study, D2: Demonstration and Simulation Plan

SpaceFibre Network Discovery and Configuration Protocol Development

Pietro Nannipieri
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa, Italy
pietro.nannipieri@unipi.it

Gionata Benelli
IngeniArs S.r.l.
Via Ponte a Piglieri 8, I-56121, Pisa, Italy
gionata.benelli@ingeniars.com

Daniele Davalle
IngeniArs S.r.l.
Via Ponte a Piglieri 8, I-56121, Pisa, Italy
daniele.davalle@ingeniars.com

Luca Fanucci
Dept. of Information Engineering
University of Pisa
Via Caruso, I-56122, Pisa, Italy
luca.fanucci@unipi.it

David Jameux
ESTEC Space Research and Technology Centre
European Space Agency - ESA
Noordwijk, The Netherlands
david.jameaux@esa.int

***Abstract*—The SpaceFibre Network Discovery & Configuration Protocol (NDCP) is intended to provide a standard mechanism to permit the detection of SpaceFibre Devices connected to a SpaceFibre (SpFi) network, which may be unknown. It is now applicable for laboratory use and, in the future, for Lunar Gateway type of modular spacecraft or the health check of known SpaceFibre Network. A SpFi Network shall be composed of a determined number of SpaceFibre Nodes and SpaceFibre Switches. Each of them will have a set of configuration parameters and status register, according to the requirements defined in the SpaceFibre standard. The paper proposes a SpaceFibre-NDCP, which builds on the packet format and semantics standardised as part of the Remote Memory Access Protocol (RMAP). It intends to extend the protocol described in the SpaceWire NDCP. Therefore, if possible, it uses the same definitions, schemes, and structures, adding the components necessary to support SpaceFibre Devices.**

***Index Terms*—SpaceWire, SpaceFibre network management, NDCP, satellite, data-handling, high-speed**

I. Introduction

In the last few years, the complexity of spacecraft grew constantly, both in the number of devices interconnected and in the bandwidth involved [1]. SpaceFibre [2] is a largely investigated candidate protocol, to address the need of future spacecraft in terms of bandwidth, reliability, and power efficiency [3]. SpaceFibre is built as the evolution of the successful SpaceWire protocol [4], and it is backwards compatible at the network level. Indeed, one of the key strengths that contributed to the success of SpaceWire was the capability to remotely control and configure [5] each of the devices available in the network thanks to the SpaceWire Network Discovery & Configuration Protocol (SpW -NDCP) [6]–[8]. The objective of this work is to propose the same upper layer protocol, capable of remotely controlling and configuring nodes, for the SpaceFibre protocol, taking inspiration from SpaceWire's successful history and working to maintain the desired retro compatibility. A future perspective is also the integration of the proposed protocol in existing mixed SpaceFibre/SpaceWire high-level simulators [9].

The SpaceFibre Network Discovery & Configuration Protocol (SpFi-NDCP) is intended to provide a standard mechanism to permit the detection of SpaceFibre devices connected to a SpaceFibre network, which may be unknown. A network shall be composed by a determined number of SpaceFibre nodes [10] and SpaceFibre switches [11]. Each of them will have a set of configuration parameters and a status register

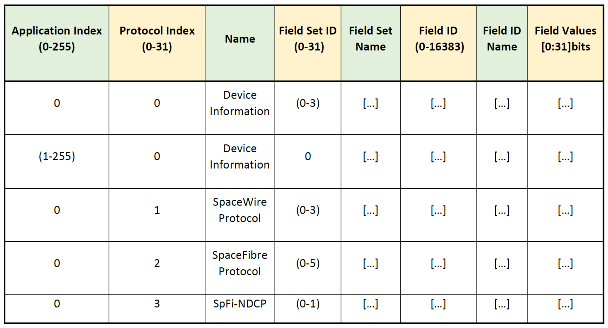
The presented work addressed the design of an NDCP protocol for SpaceFibre through the following tasks:

1) Definition of Network requirements for the complete set of status and configuration parameters required for SpaceFibre Nodes and Switches;
2) Definition of a procedure to discover and configure SpaceFibre Networks, taking also into account the needs of the SpW Network;
3) Definition of an XML Schema, which can be used to describe mixed SpaceWire-and-SpaceFibre Networks;
4) Definition of an HW Network demonstrator architecture which could be used to demonstrate the functionality.

We will refer to every single SpFi/SpW Node or Switch as a Device. The Device which will be managed by other Devices on the Network is referred to as peripheral Devices, while Nodes managing other Devices are referred to as control Devices. To allow a Control Device to access, with the aim of writing or reading, a particular register of a certain Device over the Network, the data structure of the register file of each Device shall be standardised.

II. Network Requirements

Based on the SpaceFibre standard specification, we defined network requirements for the complete set of status and configuration parameters required for SpaceFibre Nodes and Switches. A precise scheme for organising in a unique configuration space all the SpaceFibre Devices is proposed and brief indications are reported in this section. The controlling idea is to have a non-contiguous memory space to better distinguish between different Devices and, inside the same



Application Index (0-255)	Protocol Index (0-31)	Name	Field Set ID (0-31)	Field Set Name	Field ID (0-16383)	Field ID Name	Field Values [0:31]bits
0	0	Device Information	(0-3)	[...]	[...]	[...]	[...]
(1-255)	0	Device Information	0	[...]	[...]	[...]	[...]
0	1	SpaceWire Protocol	(0-3)	[...]	[...]	[...]	[...]
0	2	SpaceFibre Protocol	(0-5)	[...]	[...]	[...]	[...]
0	3	SpFi-NDCP	(0-1)	[...]	[...]	[...]	[...]

Fig. 1. Register Map General Selector

Device, between the different Ports/Virtual Channels/Lanes. Data is stored in 32 bits register named Field Values. The registers are identified by an address, composed of different fields. Such address fields have been already defined and described in SpaceWire Network Discovery and Configuration Protocol. To maintain backward compatibility, we adopted the same addressing scheme. SpFi-NDCP defines a standard set of management parameters for each Device and uses these parameters to discover and configure the SpW/SpFi Network.

A scheme for organizing in a unique configuration space all the SpaceFibre devices is proposed. Figure 1 shows the high-level structure of the register map selector.

The controlling idea is to have a non-contiguous memory space to better distinguish between different devices and, inside the same device, between the different ports/virtual channels/lanes. Data is stored in 32 bits registers named Field Values. The registers are individuated by an address, composed of different fields. Such address fields have been already defined and described in SpaceWire Network Discovery and Configuration Protocol. To maintain backward compatibility, we adopted the same addressing scheme, which is hereby recalled: The Designed configuration space for SpaceFibre status and configuration space is identified by the tuple $< ApplicationIndex; ProtocolsIndex >< 0; 2 >$. The entire space has been divided into separate Field Sets:

- Field Set ID 0: Node Information, Reserved, as specified for the SpaceWire protocol in SpaceWire NDCP standard
- Field Set ID 1: Links Status & Configuration
- Field Set ID 2: Virtual Network Status & Configuration
- Field Set ID 3: Routing Table
- Field Set ID 4: Broadcast Interface

The parameters applicable to a SpFi-NDCP packet are briefly reported here. The parameters do not change from [6], maintaining retro compatibility with SpW-NDCP capable nodes. Clauses 5.1.1, 5.1.2 and 5.1.4 to 5.1.17 of ECSS-E-ST-50-52C shall apply. These clauses define the various fields of an RMAP command or reply packet.

Application_Index

- *RMAP Equivalent:* Application Index[31:24]
- *Range:* $[0, 255]$

Specifies the application to which fields are associated for reading, writing or compare-and-swap operation. The value of the Application_Index parameter shall refer to the application by indexing into the list of supported applications provided by a peripheral device. If the value of the associated Protocol_Index parameter is zero, and the value of the Application_Index parameter is zero, fields associated with the peripheral device shall be accessed. If the value of the associated Protocol_Index parameter is zero, and the value of the Application_Index parameter is non-zero, fields associated with the corresponding application for all protocols shall be accessed. If the value of the associated Protocol_Index parameter is non-zero, and the value of the Application_Index parameter is non-zero, fields associated with the corresponding protocol for the corresponding application shall be accessed.

Current_Field_Value

- *RMAP Equivalent:* Mask
- *Range:* $[0, 2^3 2 - 1]$

Contains the expected or actual current value of a field for a compare-and-swap operation.

Field_Count

- *RMAP Equivalent:* Data Length
- *Range:* $[0, 2^1 4 - 1]$

Contains the desired or actual number of fields in a field set to be accessed in a write or read operation. The maximum number of fields a read or write operation may access is a complete field set (however, less than or equal to $2^1 4 - 1$).

Field_ID

- *RMAP Equivalent:* Address[13:0]
- *Range:* $[0, 2^1 4 - 1]$

Specifies the field, or the first field in a contiguous range, to be accessed in a write, read or compare-and-swap operation.

Field_List

- *RMAP Equivalent:* Data
- *Range:* $[0, 2^1 4 - 1]$ field values (32 bit)

Specifies the field values to be written/read in a write/read operation

FieldSet_ID

- *RMAP Equivalent:* Address[18:14]
- *Range:* $[0, 31]$

Specifies the field set to be accessed in a write, read or compare-and-swap operation

New_Field_Value

- *RMAP Equivalent:* Data
- *Range:* $[0, 2^3 2 - 1]$

Contains the desired value of a field for a compare-and-swap operation.

Peripheral_Address

- *RMAP Equivalent:* Target SpaceFibre/ SpaceWire Address
- *Range:* 0 or more SpaceWire data characters

Specifies the SpaceWire address to be used to reach a peripheral device from a control device. SpaceWire address is specified using path or regional addressing (when using SpFi/SpW-NDCP it is not possible to address a peripheral using logical addressing).

Protocol_Index

- *RMAP Equivalent:* Address[23:19]
- *Range:* $[0, 31]$

Specifies the protocol to which fields are associated for reading, writing or compare-and-swap operation. If the value of the associated Protocol_Index parameter is zero, and the value of the Application_Index parameter is non-zero, fields associated with the corresponding application for all protocols shall be accessed. If the value of the associated Protocol_Index parameter is non-zero, and the value of the Application_Index parameter is non-zero, fields associated with the corresponding protocol for the corresponding application shall be accessed.

Reply_Address

- *RMAP Equivalent:* Reply_address
- *Range:* $[0, 12]$ SpaceWire data characters

Specifies the SpaceWire address to be used to reach the control device from a peripheral device. The Reply_Address shall be specified using SpaceWire path or regional addressing. The Reply_Address parameter shall contain zero data characters when a logical address is to be used for routing the reply to a control device.

Reply_LA

- *RMAP Equivalent:* Initiator Logical Address
- *Range:* 1 SpaceWire data character

Specifies the logical address to be used to reach the control device from a peripheral device. Reply_LA contains either the logical address of the control device, if the control device has a logical address, or 0xFE otherwise.

Status

- *RMAP Equivalent:* Status
- *Range:* $[0, 255]$

Contains the status or error code of a write, read or compare-and-swap operation. Valid values for the status parameter shall be those specified in Clause 5.6 of [RD03].

Transaction_ID

- *RMAP Equivalent:* Transaction Identifier
- *Range:* $[0, 65535]$

The Transaction_ID parameter shall be used to associate request/response primitives. Two primitives related to the same transaction shall have the same Transaction_ID.

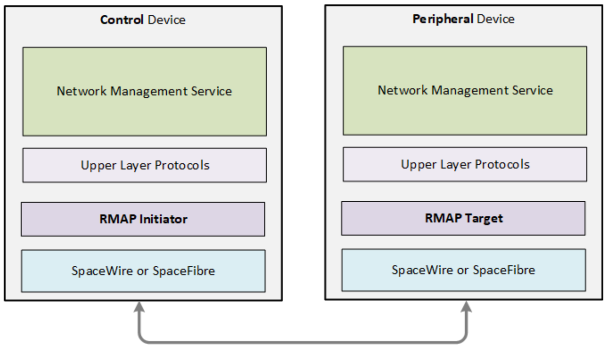


Fig. 2. SpFi-NDCP Reference Architecture, Control Vs Peripheral Devices

III. Network Discovery, Control and Configuration Procedure

The SpaceFibre-NDCP assumes a layered architecture for carrying out Network management activities. The Network management architecture consists of two layers: A Network Management Service (NMS) and a communication protocol based on RMAP. We defined a procedure to discover and configure mixed SpaceFibre and SpaceWire Networks, considering work performed for the SpaceWire NDCP standard. SpaceFibre-NDCP builds on the packet format and semantics standardised as part of the remote memory access protocol (RMAP). It intends to extend the protocol described in SpaceWire NDCP. Therefore, if possible, it uses the same definitions, schemes, and structures, adding the components necessary to support SpaceFibre devices. The reference architecture is fully compatible with the one specified section 4.2.1 in SpaceWire NDCP [6]. It is briefly illustrated here for clarity. SpFi-NDCP defines a standard set of management parameters for each device and uses these parameters to discover and configure the SpW/SpFi network. The SpaceFibre-NDCP assumes a layered architecture for carrying out network management activities. The network management architecture consists of two layers: i) A network management service (NMS) and ii) A communication protocol based on RMAP.

The NMS on a control device carries out the following operation through the supported communication protocols (SpW and SpFi):

In Figure 2, which illustrates a possible schematisation of an NMS, the network management service on the control device may access indirectly the SpFi-NDCP communications protocol. Which can be interfaced directly or using an (optional) device driver. This is up to the system designer; in our examples, we will assume that no driver is included in the application/protocol stack. Interoperability is guaranteed by placing all functionality strictly necessary for network discovery and device identification in the peripheral device. No standardisation of the control device network management service is therefore necessary. This permits devices to be manufactured supporting discovery and configuration maintaining

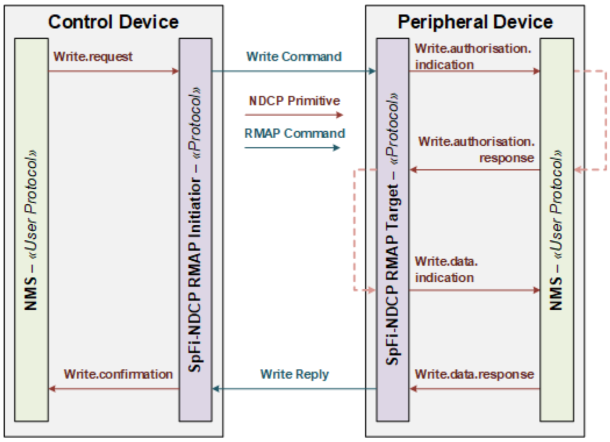


Fig. 3. SpFi-NDCP Write Operation

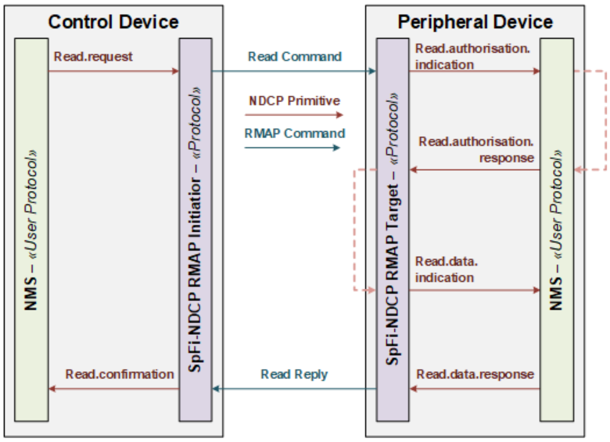


Fig. 4. SpFi-NDCP Read Operation

flexibility at the NMS level, without enforcing unnecessary standardisation. A possible definition of the Network discovery algorithm is proposed as the baseline for real application. SpFi/SpW-NDCP provides a standard mechanism for accessing peripheral device management parameters from a control device in a mixed SpFi/SpW network. The communications protocol makes use of services like the ones offered by the remote memory access protocol (RMAP). It provides three operations: Write, Read and Compare-and-Swap.

A. SpFi- NDCP Primitives

a) Write operation: permits a control device to set the value of one or more device fields. Where an operation accesses multiple fields, these are a contiguous range of field identifiers, and must all fall within the same fieldset. a. Minimum write length: 8 bytes. The SpFi-NDCP service interface shall support six primitives related to the Write operation: Control devices (Write.request, Write.confirmation), Peripheral devices (Write.authorisation.indication, Write.authorisation.response, Write.data.indication, Write.data.response). The way these primitives are used to perform a write operation is shown in Figure 3. The control device decides that it wants to write a certain field of the peripheral device: it sends a request primitive to the Controller of the RMAP initiator, which maps the request on an RMAP write command, which is sent to the peripheral device and decoded from its RMAP target. The RMAP target then generates an authorisation.indication primitive, to be sent to the NMS to understand if the Write operation is authorised or not. The NMS provides the authorisation.response, and in case the write operation is authorised, the data.indication primitive will provide to the NMS the exact field and field data to be written. Finally, the NMS will generate a data.response with the outcomes of the write operation, which will be then sent from the RMAP target as a write reply to the control device RMAP Initiator. The RMAP initiator then will convert the information through the confirmation primitive and will send that primitive to its NMS.

b) Read operation: permits the control device to access the value of one or more device fields. Where an operation accesses multiple fields, these are a contiguous range of field identifiers, and must all fall within the same fieldset. a. Minimum read length: 64 bytes. The read operation is performed by a control device to read one or more fields of a peripheral device. As already specified for the write operation, the SpFi-NDCP control device shall include an RMAP Initiator and the peripheral device shall include an RMAP target. Compliance with the read operation specified in [6] is guaranteed. The following clauses from the RMAP standard shall apply 5.4.1,5.4.2, 5.4.3.1 to 5.4.3.3, 5.4.3.5 to 5.4.3.13 and 5.4.3.4.2 to 5.4.3.4.9. The SpFi-NDCP service interface shall support six primitives related to the Read operation: Control devices (Read.request, Read.confirmation), Peripheral devices (Read.authorisation.indication, Read.authorisation.response, Read.data.indication, Read.data.response). The way these primitives are used to perform a write operation is shown in Figure 4: The control device decides that it wants to read a certain field of the peripheral devices: it sends a request primitive to the Controller of the RMAP initiator, which maps the request on an RMAP read command. The command is then sent to the peripheral device and decoded by its RMAP target, which will generate an authorisation.indication primitive, to be sent to the NMS to understand if the read operation is authorised or not. The NMS provides the authorisation.response, and in case the read operation is authorised, the data.indication primitive will provide to the NMS the exact fields to be read. Finally, the NMS will generate a data.response with the outcomes of the read operation (including reading fields), which will be sent from the RMAP target as a read reply to the control device RMAP Initiator. Finally, the RMAP Initiator will convert the information through the confirmation primitive and will send that primitive to its NMS.

c) Compare-and-Swap operation: requests that the peripheral device write the value of a field, only if the current value of that field matches some known value. The peripheral device must therefore read the field, compare it to the specified value and, only if there is a match, write the new value of the field. These read and write operations must be conducted atomically by the peripheral device, to resolve contention between multiple control devices. The Compare and Swap (CAS) operation can be invoked by a control device, to set the value in the peripheral device of a single field, in case the current value of that field is equal to the know and specified value. This operation is quite complex and made up of several operations. In the peripheral device, the following operations are to be carried out atomically (no operation can be carried out on fields in between): Read the current value of the specified field; Compare it with a specified reference value; Write the new value and reply indicating successful operation if the values are the same, otherwise do not write and reply indication operation failure. All the CAS-related primitives rely on the RMAP protocol. All 5.5 subclauses (except from 5.5.3.4.1) of [5] shall apply. The RMAP structure on which the CAS operation relies is the read-modify-write. The compare operation is done so that data are written in the peripheral device only if the read data matches the data specified in the mask field of the read-modify-write RMAP operation. The data to be written is to be specified in the data field of the read-modify-write command. The following primitives are specified: Control devices (CAS.request, CAS.confirmation); Peripheral devices (CAS.authorisation.indication, CAS.authorisation.response, CAS.read.indication, CAS.read.response, CAS.write.indication, CAS.write.response). The way these primitives are used to perform a write operation is shown in Figure 5:

The control device decides that it wants to perform a CAS operation on the peripheral devices: it sends a request primitive to the controller of the RMAP initiator, which maps the request on an RMAP read-modify-write command. The command is sent to the peripheral device and decoded from its RMAP target, which then generates an authorisation.indication primitive, to be sent to the NMS to understand if the CAS operation is authorised or not. The NMS provide the authorisation.response, and in case the CAS operation is authorised, the read of the data to be compared is required through the read.indication primitive. The read data is sent back through the read.response primitive and is compared with the reference value. If the outcome is positive, then the protocol sends to the NMS a write.indication primitive. The NMS finally will generate a data.response with the outcomes of the write operation, which will be then sent from the RMAP target as a read-modify-write reply to the control device RMAP Initiator. Finally, the RMAP initiator will convert the information through the confirmation primitive and will send that primitive to its NMS.

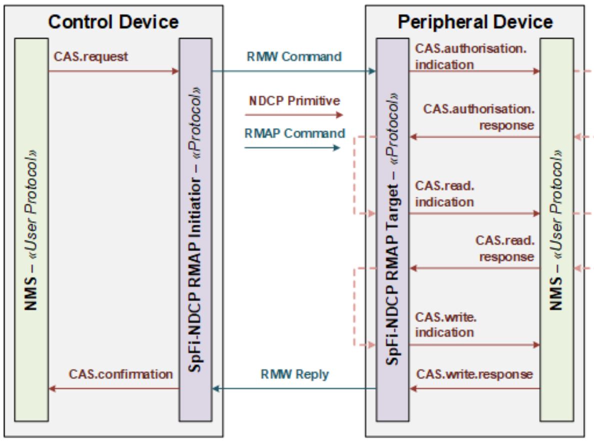


Fig. 5. SpFi-NDCP Compare and Swap Operation

B. Device Identification

The first operation that a control device shall perform is to identify all the nodes of the network. To do so, it needs to perform a read operation on all fields of the Device Identification fieldset. In doing so, the control device will be informed of the device type (with version information), the number of links on the device, which of those links are active, whether the device has already been identified and controlled by another control device and the device identifier.

C. Network changes

In the case that a new device is plugged into the network, an inactive link is activated at some point, or a device is disconnected and then reconnected to the network, the Device ID field is not necessarily known by the control device. This is the case mostly for network debugging. It is assumed that once the network has been deployed, its behaviour is expected to be static. Considering that, there are two options for the SpFi-NDCP. The first one is that periodically the Control nodes perform a discovery routine to understand if the network has a new node connected to it. This approach is easy to be implemented and does not require any effort on the peripheral devices. An alternative approach may be to request to each peripheral device send a specific Broadcast message which can be used to let the control device know that it shall perform a discovery operation. This approach is far more efficient as it does not require any period of network discovery; on the other hand, it supports only SpaceFibre nodes (SpaceWire nodes are not allowed to send broadcast messages) and requires extra control on the peripheral device.

D. Multiple Control Devices

The control device, owning a peripheral device by assigning it the Device ID can be identified using the Ownership Port, Owner Address and Owner Logical Address fields. Using this information, a control device may understand if a peripheral device is owned by another control device and if this ownership is valid. The validity of the current owner can be determined if the owning control device is also a reachable peripheral device. In any case, the policies governing the way networks are designed and discovered, the way that devices are claimed and the rules for possible competing control devices must be decided appropriately for the target mission or application scenario.

E. Network Discovery

To fully discover the network, the control device shall perform Device identification on all the devices connected to the network via an active link. The control device does not know a priori the network topology; therefore, it must explore the network, which can be schematized as a tree structure. The approach here can be defined by the user. General approaches may be to explore the tree structure either breadth-first or depth-first. These two approaches are compared in terms of computational effort. We chose arbitrarily to take the depth-first case as an example. The operation that the Control device shall carry out are summarized here:

1) The control device shall identify which of its ports are active and running. Let us identify the active port number with i, ranging from N to M.
2) Then, the control device will identify the device (identified as Node K) Connected to port i, reading its device identification information. In particular, it will rea if the device has already been identified (if not, e.g. the device ID is set to the default value, it will identify it by giving it a unique identifier through a compare-and-swap operation) and if the device has other ports connected to other nodes of the network.
3) In case there are further devices connected to Node K, we repeat the operation described in 2) with all the nodes connected to node K. Otherwise, if there are no more ports to be discovered, we will go backwards in the hierarchical level and increase the port number i.

In realistic SpaceFibre networks not all the valid links in the network are necessarily active (i.e. running) at any given time (i.e. at discovery time). The network discovery algorithm as it has been described above will detect only active links. Depending on the application scenario, it may be desirable for the control device to determine the full network topology, considering also not currently active links. To do so, during the discovery process described above, the control device shall appropriately wake up the non-active nodes (the procedure is different depending if the node is a SpaceFibre or SpaceWire node), discover them and then put them back in its idle status. In the following, the Network discovery operation is carried out on an example network.

F. Example Network

In Figure 6 a diagram of a small representative network is shown. It has been chosen to illustrate step by step how the network discovery algorithm based on SpFi-NDCP primitives could work.

Each box in the diagram is a device (it is indicated if it is a control or peripheral). The devices are then interconnected with both SpaceFibre and SpaceWire links (both illustrated with a red arrow as it does not change anything from a network discovery perspective). The red lines interconnecting the devices are the communication links, and the numbers in blue represent the port numbers. In the following, the analysis step-by-step of the discovery procedure is carried out. In Figure 7 the discovery steps are shown.

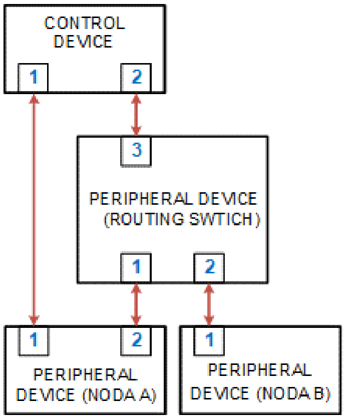

CONTROL DEVICE
1
2
3
PERIPHERAL DEVICE (ROUTING SWTICH)
1
2
1
2
PERIPHERAL DEVICE (NODA A)
1
PERIPHERAL DEVICE (NODA B)

Fig. 6. SpFi-NDCP Representative Network

First, the control device will identify its active ports, 1 and 2. Then (a) it will identify the device at the end of link 1, reading the device information of node A, and its active ports, checking that it has not been identified previously and consequently setting its device ID to 1. Next (b) the same operation has been carried out on port 2 of the control device since the discovered device is a node and not a switch. Therefore, the control device will identify the device connected to port 2, which will be discovered to be a routing switch. The control device will identify it with device ID 2 and recognise that it has 2 active ports to continue the discovery. The next step (c) will discover that port 2 of the routing switch is connected to port 2 of Node A, which has already been discovered. Finally (d) the control device will discover node B as the node connected to port 2 of the routing switch, and after checking that it has not been previously discovered and owned by someone else, it will assign its device ID 3. Of course, a real network will be far more complex, but the same algorithm can be applied virtually to any network. However, the system designer will need to pay attention to a few corner cases where they shall define their procedures. Those cases are briefly illustrated in the following.

IV. XML SCHEMA

Based on the outputs of the previous task, an XML Schema was defined, to describe mixed SpFi and SpW networks in a human-readable language such as XML, composed of NDCP-capable devices. We have implemented a flexible and complete schema, capable of handling devices which are not NDCP aware, to ensure higher flexibility and simplify the application of this standard. The highest-level element (root element), DataHandlingSystem, can contain several networks. This offers the possibility of describing particular configurations with networks sharing nodes between them. A Network element describes a complete SpFi, SpW or mixed network. The network is described as an acyclic graph, e.g., as a collection

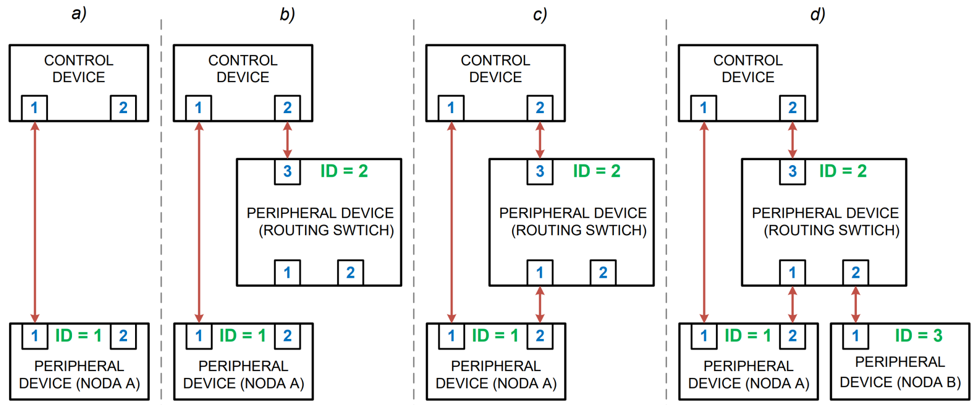

a)
b)
c)
d)
CONTROL DEVICE
1
2
ID = 2
3
PERIPHERAL DEVICE (ROUTING SWTICH)
ID = 1
PERIPHERAL DEVICE (NODA A)
ID = 3
PERIPHERAL DEVICE (NODA B)

Fig. 7. **SpFi-NDCP Network Discovery**

of nodes and links between them. A Network element can contain any number of SpWNode, SpFiNode, SpWSwitch, SpFiSwitch, SpWLink and SpFiLink elements. The full XML schema is omitted due to the limited length of the paper but is available on request.

V. Proposed Network Demonstrator Architecture

We developed a hardware demonstrator architecture to be used in the next steps to demonstrate SpaceFibre NDCP features. A possible set-up of the network is described in [12]: it is composed of a routing switch with 4 SpaceFibre ports; port 1 is connected to an on-board computer, port 2 with a high bandwidth instrument, port 3 to a channel which mixes up SpaceFibre and SpaceWire traffic coming from a medium bandwidth instrument and port 4 is connected to mass memory. Interconnection within the routing switch is done exploiting the virtual network mechanism, a feature of the SpaceFibre network layer: each tuple ¡Port-VC¿ is part of a network of 2 or more ends, real-time configurable by the user through RMAP commands.

A. Hardware Resources

The **routing switch** that we propose can be provided by IngeniArs [11] and mapped on a commercial Xilinx Ultrascale+ ZCU102 board, equipped with appropriate high-speed serial connectors thanks to the ALDEC SATA FMC module. The SpaceFibre ports are implemented with the IngeniArs SpaceFibre CoDec IP [10], which has been deeply tested and validated on-field for years and supports all the protocol stack up to the lower part of the lane layer [13]. The codec is multilane capable [14] and also compatible with a reduced version of the SpaceFibre standard [15] A key feature of the routing switch is that it can be instantiated with a generic number of ports, each one with a generic number of VCs.

SpaceART (SpaceWire/SpaceFibre Analyser Real-Time) [16], [17], is a complete testing solution for high-speed links in space applications. SpaceART supports both SpaceWire and SpaceFibre standards. And can be used as **mass memory** emulator, also compliant with a PXI interface [18]. It operates as an SpW/SpFi EGSE (Electrical Ground Segment Equipment), generating, processing and consuming SpW/SpFi packets in real-time, allowing the validation of SpFi/SpW-based devices at their full bandwidth. SpaceART is also an SpW/SpFi link analyser, allowing to monitor the link status. In both operation modes, error injection is feasible (both on the Rx and Tx side) allowing to stimulate appropriately a wide range of error situations. The **high bandwidth instrument** can be emulated employing IngeniArs SpaceFibre IP core, implemented on an FPGA development kit or also on the same SpaceART used as mass memory, for cost reduction. SpaceART unit handles SpW to SpFi bridging. The **on board computer** can also be emulated using SpaceART. The **SpaceWire traffic** can be generated and sent over the network by the SpaceART.

B. Minimal hardware setup for NDCP demonstrator

Figure 8 shows the minimal hardware setup for the SpaceFibre NDCP described in the previous sections. The setup was conceived to minimize the cost; therefore the units were com-

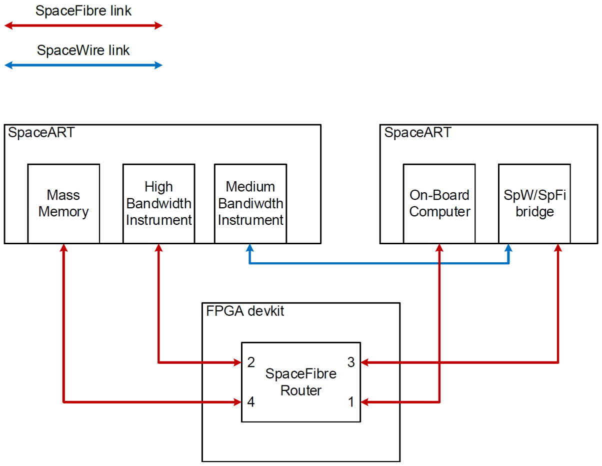


Fig. 8. Hardware setup for NDCP demonstrator

pacted into the least possible number of SpaceART units. The total number of hardware parts needed is 2x SpaceART units, SpaceFibre and NDCP capable and 1 FPGA development kit with related FMC board to implement 4x SpaceFibre interfaces. This FPGA devkit will host the IngeniArs SpaceFibre router IP core.

VI. Conclusions

In this work, we proposed a standard method to perform network discovery and configuration on a mixed SpaceWire - SpaceFibre satellite data-handling network. We briefly reported how the configuration space of every single node/device in the network shall be organised to fully support the design protocol and to be retro-compatible with the existing SpW-NDCP. The method to remotely discover and configure t a network with a master node is described with examples, and we propose a representative network that may be used for future validation of the proposed protocol. Such an advancement on the upper layer protocols for the SpaceFibre network will contribute to the uptake of the technology itself, which is becoming more and more mature for future employment on a wide set of satellite missions.

Acknowledgment

This work has been funded by the European Space Agency under contract no. 4000126296/18/NL/AF.

References

[1] B. Evans, P. Thompson, G. Corazza, A. Vanelli-Coralli, and E. Candreva, “1945-2010: 65 years of satellite history from early visions to latest missions,” *Proceedings of the IEEE*, vol. 99, no. 11, pp. 1840–1857, 2011.

[2] European Cooperation for Space Standardisation, *SpaceFibre standard, ECSS-E-ST-50-11C*. European Cooperation for Space Standardisation.

[3] R. Xiong, L. Li, X. Yi, C. Zhang, G. Yang, and H. Fang, “The multilevel dynamic bandwidth allocation and performance analysis of spaceborne network based on spacefibre,” *Proceedings of the International Astronautical Congress, IAC*, vol. 2018-October, 2018.

[4] European Cooperation for Space Standardisation, *SpaceWire – Links, nodes, routers and networks, ECSS-E-ST-50-12C*. European Cooperation for Space Standardisation.

[5] ——, *SpaceWire – Remote memory access protocol, ECSS-E-ST-50-52C*. European Cooperation for Space Standardisation.

[6] ——, *SpaceWire Network Discovery Configuration Protocol standard, ECSS-E-ST-50-54*. European Cooperation for Space Standardisation.

[7] K. Romanowski, P. Tyczka, W. Holubowicz, R. Renk, V. D. Kollias, N. Pogkas, and D. Jameux, “Spacewire network management using network discovery and configuration protocol: Spacewire networks and protocols, short paper,” 2016, Conference paper.

[8] A. Tavoularis, V. Vlagkoulis, F. Kostopoulos, T. Le Ngoc, B. Dellandrea, L. Fossati, J. Ilstad, and D. Jameux, “Spacewire components, long paper: An ip core for the spw family of protocols,” 2016.

[9] A. Leoni, P. Nannipieri, D. Davalle, L. Fanucci, and D. Jameux, “Shine: Simulator for satellite on-board high-speed networks featuring spacefibre and spacewire protocols,” *Aerospace*, vol. 6, no. 4, 2019.

[10] P. Nannipieri, G. Dinelli, A. Marino, L. Dello Sterpaio, A. Leoni, L. Fanucci, and D. Davalle, “A serial high-speed satellite communication codec: Design and implementation of a spacefibre interface,” *Acta Astronautica*, vol. 169, pp. 206–215, 2020.

[11] A. Leoni, P. Nannipieri, and L. Fanucci, “Vhdl design of a spacefibre routing switch,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E102A, no. 5, pp. 729–731, 2019.

[12] P. Nannipieri, L. Fanucci, and F. Siegle, “A representative spacefibre network evaluation: Features, performances and future trends,” *Acta Astronautica*, vol. 176, pp. 313–323, 2020.

[13] P. Nannipieri, D. Davalle, and L. Fanucci, “A novel parallel 8b/10b encoder: Architecture and comparison with classical solution,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E101A, no. 7, pp. 1120–1122, 2018.

[14] G. Dinelli, P. Nannipieri, A. Marino, L. Fanucci, and L. Dello Sterpaio, “The very high-speed spacefibre multi-lane codec: Implementation and experimental performance evaluation,” *Acta Astronautica*, vol. 179, pp. 462–470, 2021.

[15] G. Dinelli, P. Nannipieri, D. Davalle, and L. Fanucci, “Design of a reduced spacefibre interface: An enabling technology for low-cost spacecraft high-speed data-handling,” *Aerospace*, vol. 6, no. 9, 2019.

[16] A. Marino, A. Leoni, L. D. Sterpaio, P. Nannipieri, G. Dinelli, G. Benelli, D. Davalle, and L. Fanucci, “Spaceart spacewire and spacefibre analyser real-time,” in *2020 IEEE International Workshop on Metrology for AeroSpace, MetroAeroSpace 2020 - Proceedings*, 2020, pp. 244–248.

[17] L. D. Sterpaio, A. Marino, P. Nannipieri, G. Dinelli, D. Davalle, and L. Fanucci, “A complete egse solution for the spacewire and spacefibre protocol based on the pxi industry standard,” *Sensors (Switzerland)*, vol. 19, no. 22, 2019.

[18] L. D. Sterpaio, P. Nannipieri, A. Marino, and L. Fanucci, “Design of a spacewire/spacefibre egse system based on pxi industry standard,” in *2019 IEEE International Workshop on Metrology for AeroSpace, MetroAeroSpace 2019 - Proceedings*, 2019, pp. 22–26.

SpaceFibre and SpaceWire Network Management: NDCP Version 2

Networks and Protocols, Long Paper

Jarosław Kwiatkowski, Krzysztof Romanowski*, Piotr Tyczka
ITTI Sp. z o.o.
Poznań, Poland
{Jaroslaw.Kwiatkowski,Krzysztof.Romanowski,Piotr.Tyczka}
@itti.com.pl

David Jameux
ESTEC
European Space Agency
Noordwijk, The Netherlands
David.Jameux@esa.int

Abstract—**The SpaceFibre standard defines the Management Information Base (MIB), a repository of parameters used for configuring, controlling, and monitoring the operation of a SpaceFibre device, as well as the MIB service, which provides an interface for manipulating the values of the parameters. The standard specifies the Remote Memory Access Protocol (RMAP) as the means of remote management of SpaceFibre networks. This paper presents an alternative approach of adopting a dedicated network management protocol based on the Network Discovery and Configuration Protocol that had been proposed for SpaceWire networks (SpW-NDCP), extended so as to handle SpaceWire, SpaceFibre, and mixed networks, called NDCP version 2. Details of the new protocol are shown and compared to the original SpW-NDCP. An accompanying XML network representation is proposed. A demonstrator mixed SpaceWire/SpaceFibre network is described, including a custom-made SpaceFibre node built on a system-on-a-chip and supporting the NDCP v.2.**

Keywords—network management, NDCP, SpaceFibre, SpaceWire

I. Introduction

The ever-growing complexity of on-board data handling systems requires supporting tools for network management. The 2008 edition of the SpaceWire standard [1] did not address network management explicitly. Its current revision [2] as well as the SpaceFibre standard [3] both define the Management Information Base (MIB), a repository of parameters used for configuring, controlling, and monitoring the operation of a SpaceWire or SpaceFibre device, as well as the MIB service, which provides an interface for manipulating the values of the parameters.

While the SpaceWire standard does not suggest any particular protocol for the MIB service, its SpaceFibre counterpart specifies the Remote Memory Access Protocol (RMAP) [4] as the means of remote management of SpaceFibre networks. This implies the management parameters are accessed at specific memory locations of the managed device. However, the memory addresses as well as the ranges and formats of many of the parameters are not standardized and can differ between device models, like what can be found with RMAP usage in SpaceWire, where not only the parameter values and addresses, but also the functionality, the addressing units, and the byte order depend on the specific device model.

An initial idea of a unified memory space specification was presented in [5] as part of a proposed new transaction layer of SpaceFibre. That specification can facilitate RMAP-based remote management and was adopted for a SpaceFibre routing switch proposed in [6].

RMAP is not the only possible option for accessing network management functionality. An alternative – a dedicated management protocol – was proposed for SpaceWire networks. Initially called Plug-and-Play [7], it was later renamed the Network Discovery and Configuration Protocol (SpW-NDCP) [8]. It was implemented in IP cores (e.g. [9,10]) as well as in commercially available chips (e.g. [11,12]). A network management tool called SPACEMAN was developed that makes use of the protocol for SpaceWire network management [13,14].

This paper proposes a management protocol based on the SpW-NDCP, generalized so as to be applicable to SpaceWire, SpaceFibre, and mixed SpaceWire/SpaceFibre networks. Extending SpW-NDCP to the domain of SpaceFibre was the primary objective of the recently completed ESA-funded project *FiMan*. The extended protocol is termed NDCP version 2. In remainder of the paper this protocol is referred to as NDCP v.2, while the original version is denoted SpW-NDCP.

Section II presents the general principles of the protocol. The mapping between the parameter space of NDCP v.2 and the MIBs specified in the SpaceFibre standard and in the revised SpaceWire standard are shown in Section III. Besides the MIBs, additional parameters proposed in [5] and in the *FiMan* project are included. Section IV proposes an XML format for representation of SpaceFibre, SpaceWire, and mixed networks. A mixed SpaceWire/SpaceFibre network that was used for demonstrating NDCP v.2-based network discovery and configuration, with a SpaceFibre node that supports the new protocol, are described in Section V. Finally, Section VI presents conclusions.

II. Principles

The SpW-NDCP protocol adopted as the base for developing its new version provides a standard mechanism for accessing device management parameters. Device information is held in fields of 32 bits and each field has an identifier. Related fields are grouped together into field sets. There are field sets for each supported protocol, application, and application protocol use. The protocol makes use of the frame format defined for the RMAP and offers three operations: write, read, and compare-and-swap (CAS), targeting SpW-NDCP field identifiers rather than memory addresses (unlike RMAP). To identify a field as a part of a

This work has been funded by the European Space Agency under contract no. 4000125356/18/NL/Cbi/md.

**Corresponding author*

read, write, or CAS operation, four values need to be specified: the application (service) index, the protocol index, the field set identifier, and the field identifier. The protocol identifier (in the sense of [15]) proposed (but not standardized yet) for SpW-NDCP is 3.

The NDCP v.2 follows the packet structure, syntax, and semantics of the original SpW-NDCP. A number of extensions and changes were introduced in order to support SpaceFibre networks. The changes affect also the SpaceWire part. However, the format of the protocol frame, including the proposed protocol ID remains unchanged. In order to differentiate between the protocol versions, so that the same application can be used for handling them both, one of the NDCP v.2 fields that were reserved in the SpW-NDCP (specifically, byte 0 of the Version field of the Device Identification field set) now holds the NDCP protocol version number in the revised protocol. This is assumed to be backward-compatible, since the reserved fields are specified by the SpW-NDCP draft standard as readable and returning zero when read. The value of zero, which is supposed to be returned by a SpW-NDCP-compliant device, is interpreted as indication of NDCP version number 1, i.e. the SpW-NDCP; other values are interpreted directly as NDCP version numbers (with ‘1’ also interpreted as version 1 for simplicity), with the exception of the value ‘52’, which was found to be returned by some of the prototype SpW-NDCP-aware devices [9] instead of the expected zero.

III. NDCP FIELDS

The management parameters mapped to NDCP v.2 fields come from the SpaceWire and SpaceFibre standards, the SpW-NDCP draft standard, propositions presented in [5], and propositions originated in the *FiMan* project. The fields form a hierarchy of the following levels, with the corresponding NDCP addressing indices:

- groups, which correspond to pairs composed of an application index and a protocol index,
- field sets, which correspond to field set identifiers,
- optionally: field subsets of up to 3 levels (for port parameters); these do not have explicit corresponding addressing identifiers and only used for naming related adjacent fields,
- fields, which correspond to field identifiers.

Although neither the SpaceFibre nor the SpaceWire standards refer to network devices having both SpaceFibre and SpaceWire ports (in the same device), such heterogeneous devices are in fact produced and used (cf. the devices used for the demonstrator described in Section V). Therefore the layout of NDCP v.2 fields describing a device is basically common to SpaceWire and SpaceFibre and is organized as follows:

- Device Information group; this is almost the same as in the original SpW-NDCP and is common to SpaceWire and SpaceFibre devices:
    - Device Identification field set (the NDCP protocol version number is held in this set, as described in Section II),
    - Vendor/Product Strings field set,
    - Protocol Support field set,
    - Application Support field set;
- SpaceWire Protocol group, which includes also SpaceFibre-related parameters:
    - Device Configuration field set,
    - Port Configuration field set; these two field sets include fields that are common to SpaceWire and SpaceFibre devices as well as fields that are technology-specific, i.e. different for SpaceWire and SpaceFibre; moreover, the SpaceWire-specific fields are different than in the original SpW-NDCP,
    - Switching Table field set; this field set is almost the same as in the original SpW-NDCP, with the addition of the ‘Multicast enabled’ bit (in place of a bit that was reserved in the SpW-NDCP) for each logical address,
    - Time-code Generation field set; for SpaceWire, this field set is the same as in the original SpW-NDCP; for SpaceFibre, it is reserved (not used);
- NDCP Protocol group; this group is the same as in the original SpW-NDCP and is common to SpaceWire and SpaceFibre devices:
    - Protocol Information field set;
- Network Management Service group; this group is also the same as in the original SpW-NDCP and common to SpaceWire and SpaceFibre devices:
    - Service Information field set.

The structure of the Device Configuration field set and of each of up to 32 Port subsets of the Port Configuration field set is technology-specific. The first field in the Device Configuration field set is the Device Type, composed of the device type code and the field set structure version number. The first field in the Port field subset (associated with a single port) is the Port Type, composed of the port type code and the field subset structure version number. By using this convention a network manager application can identify the type (SpaceWire or SpaceFibre) of each port and interpret the fields correctly. It can also identify the type of the device as a whole and correctly interpret the device-level fields, which hold device-level parameters defined in either SpaceWire or SpaceFibre standard.

For a SpaceFibre port its Port field subset is further divided into field subsets for:

- Port-level Parameters,
- Virtual Channels,
- Lanes.

The Virtual Channels field subset holds next-level field subsets for each of up to 32 virtual channels. Similarly, the Lanes field subset holds next-level field subsets for each of up to 16 lanes.

The hierarchy of the NDCP fields is shown in Table 1.

TABLE I. FIELD HIERARCHY

Group	Field set	Field subset – level 1	Field subset – level 2	Field subset – level 3
Device Information	Device Identification			
	Vendor/ Product Strings			
	Protocol Support			
	Application Support			
SpaceWire Protocol	Device Configuration			
	Port Configuration	Port 0[a]		
		Port 1	Port-level parameters	
			Virtual Channels	VC0[b]
				VC1
				…
				VC31
			Lanes	Lane 0
				Lane 1
				…
				Lane 15
		…		
		Port 31	Port-level parameters	
			Virtual Channels	VC0
				VC1
				…
				VC31
			Lanes	Lane 0
				Lane 1
				…
				Lane 15
	Switching Table			
	Time-code Generation			
NDCP Protocol	Protocol Information			
Network management service	Service information			

[a] Port 0 is the configuration port

[b] VC= Virtual Channel

🟨 Common to SpaceWire/SpaceFibre

🟧 Partially common to SpaceWire/SpaceFibre

🟩 SpaceWire only

🟦 SpaceFibre only

This field hierarchy places logical description units (Device Information, Protocol, Device Configuration, Port Configuration) at levels above technology-specific (SpaceWire/SpaceFibre) subdivision. An alternative approach: placing the technology-specific division on top, i.e. having separate Protocol group for each technology with all Port field subsets in of the same type in a group, although simple for devices with only one type of ports (either SpaceWire or SpaceFibre), is less fit for devices that contain ports of both types with common numbering and a single routing (switching) table.

The full listing of the proposed NDCP fields is available as a deliverable of the *FiMan* project. An excerpt of the Device Configuration and the Port Configuration parameters for SpaceFibre devices is shown in Table II as an example. Actual NDCP fields are composed of the parameters, packing short parameters into a common field (32-bit word) where possible and avoiding mixing read-only with read-write parameters in the same word.

IV. XML REPRESENTATION

The XML network representation proposed earlier in the context of developing the SPACEMAN network management tool and using the SpW-NDCP [14], was revised and extended in the *FiMan* project, so as to accommodate all entities and parameters relevant to SpaceFibre and the NDCP v.2, while also covering SpaceWire use new attribute of a network device model (node or switch) was introduced in order to mark whether the device supports the NDCP v.2. New elements were introduced, representing entities related to SpaceFibre and to new NDCP fields. The names of some elements or attributes that were already present in the previous version were changed, reflecting the editorial changes that were introduced in the meantime in the NDCP draft [8] (e.g. changing the word 'link' to 'port' for the 'Port Information' field and its subfields like 'Return Port', which used to be called 'Link Information' field, and 'Return Link', respectively). Also, some intermediate level container elements were introduced, reflecting the hierarchy of NDCP fields.

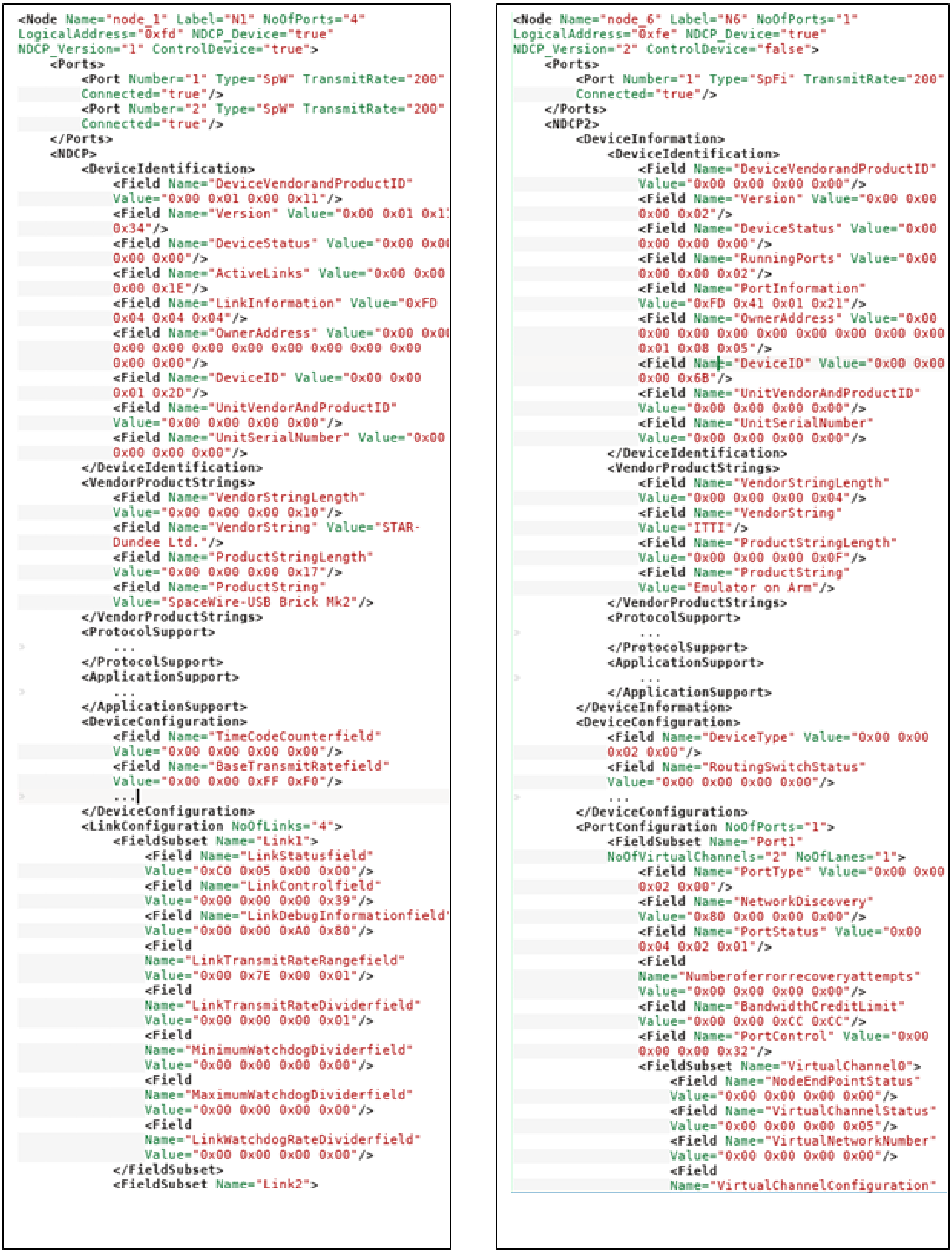
It should be noted that the proposed XML representation allows SpaceWire, SpaceFibre, and mixed SpaceWire/SpaceFibre devices, supporting either of the NDCP version (or even none at all), to coexist as parts of the same model. Two excerpts from such a single model are presented in Fig. 1. The first represents a STAR-Dundee SpW-USB Brick Mk2 with support for the SpW-NDCP; the second – the SpaceFibre SoC-based node with support for the NDCP v.2, which is described in Section V.

V. DEMONSTRATOR

The functionality of the NDCP v.2 was validated and demonstrated on several physical networks. The SPACEMAN network management tool was expanded so as to support the new protocol and to be able to connect to SpaceFibre networks. In order to have network devices with the NDCP v.2 support, two types of network nodes were developed. One is a software NDCP emulator based on a PC connected to a SpaceFibre device via a non-SpaceFibre/SpaceWire link (Ethernet or USB). The SpaceFibre device together with the PC operate as a single SpaceFibre node from the point of view of the network manager. The other type is a system-on-chip-based (SoC) node, where an existing SpaceFibre IP core is extended with an on-chip implementation of the NDCP v.2.

Table II. NDCP parameters for SpaceFibre (excerpt)

(sub)field name	common for SpW /SpFi?	present in:		width (bits)	range	unit	information held	reset value	read-only?	static?	source	notes
		node	switch									
Device Configuration field set												
Device Type	Y	Y	Y	32	0-max	n/a	bits 8-31: device type code bits 0-7: field set definition version		RO	Y	ITTI	
Routing Switch Status	Y	N	Y	32	bit map	n/a	TBD		RO	N	[3] Tab.5-37	Format not in [3]
Broadcast time-out interval	N	Y	Y	32	0-max	clock cycle or µs (TBD)	time; if µs, then range is 0 to ca. 4295 s with minimal increment 1 µs		RW	N	[3] Sect.5.8.12.2	Range and unit not in [3]
Broadcast Channel association valid	N	Y	Y	1	0-1	n/a	1: device associated with Broadcast Channel 0: not associated with any broadcast channel	0 (TBD)	RW	N	[3] Sect.5.8.12.1	
Broadcast Channel	N	Y	Y	8	0-255	n/a	broadcast channel number associated with the device	0 (TBD)	RW	N	[3] Sect.5.8.12.1	
Invalid output port error	N	N	Y	1	0-1	n/a	status flag		RO	N	[3] Sect.5.8.8.3k	May be part of Routing Switch Status.
Port Configuration field set												
Port Type	Y	Y	Y	32	0-max	n/a	bits 8-31: port type code bits 0-7: field set structure version		RO	Y	ITTI	
Network Discovery	Y	Y	Y	1	0-1	n/a	attribute flag	1: use port for discovery; 0: don't	RO	Y	[8]	
Number of Virtual Channels	N	Y	Y	5	0-31	virtual channel	actual number of virtual channels=value+1		RO	Y	ITTI	
Number of Lanes	N	Y	Y	4	0-16	lane	actual number of lanes=value+1		RO	Y	ITTI	
16-bit CRC error	N	Y	Y	1	0-1	n/a	status flag		RO	N	[3] Tab.5-37	
Frame Error	N	Y	Y	1	0-1	n/a	status flag		RO	N	[3] Tab.5-37	
CRC-8 error	N	Y	Y	1	0-1	n/a	status flag		RO	N	[3] Tab.5-37	
Sequence error	N	Y	Y	1	0-1	n/a	status flag		RO	N	[3] Tab.5-37	
Error recovery buffer empty	N	Y	Y	1	0-1	n/a	status flag		RO	N	[3] Tab.5-37	
Number of error recovery attempts	N	Y	Y	32	0-max	attempt	counter		RO	N	[3] Tab.5-37	Range not in [3]; [5] uses 6 bits
Link Reset Caused by Protocol Error	N	Y	Y	1	0-1	n/a	status flag		RO	N	[3] Tab.5-37	
Far-End Link Reset	N	Y	Y	1	0-1	n/a	status flag		RO	N	[3] Tab.5-37	
Alignment State	N	Y	Y	2	0-2	n/a	one of 3 states		RO	N	[3] Tab.5-37	'Reserved' for single-lane ports
Bandwidth Credit Limit	N	Y	Y	32	0-max	word	limit on the number of words	imple-mentation	RW	N	[3] Tab.5-36	Range not in [3]; [5] uses 32 bits as well



```
<Node Name="node_1" Label="N1" NoOfPorts="4"
LogicalAddress="0xfd" NDCP_Device="true"
NDCP_Version="1" ControlDevice="true">
    <Ports>
        <Port Number="1" Type="SpW" TransmitRate="200"
        Connected="true"/>
        <Port Number="2" Type="SpW" TransmitRate="200"
        Connected="true"/>
    </Ports>
    <NDCP>
        <DeviceIdentification>
            <Field Name="DeviceVendorandProductID"
            Value="0x00 0x01 0x00 0x11"/>
            <Field Name="Version" Value="0x00 0x01 0x1
            0x34"/>
            <Field Name="DeviceStatus" Value="0x00 0x0
            0x00 0x00"/>
            <Field Name="ActiveLinks" Value="0x00 0x00
            0x00 0x1E"/>
            <Field Name="LinkInformation" Value="0xFD
            0x04 0x04 0x04"/>
            <Field Name="OwnerAddress" Value="0x00 0x0
            0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
            0x00 0x00"/>
            <Field Name="DeviceID" Value="0x00 0x00
            0x01 0x2D"/>
            <Field Name="UnitVendorAndProductID"
            Value="0x00 0x00 0x00 0x00"/>
            <Field Name="UnitSerialNumber" Value="0x00
            0x00 0x00 0x00"/>
        </DeviceIdentification>
        <VendorProductStrings>
            <Field Name="VendorStringLength"
            Value="0x00 0x00 0x00 0x10"/>
            <Field Name="VendorString" Value="STAR-
            Dundee Ltd."/>
            <Field Name="ProductStringLength"
            Value="0x00 0x00 0x00 0x17"/>
            <Field Name="ProductString"
            Value="SpaceWire-USB Brick Mk2"/>
        </VendorProductStrings>
        <ProtocolSupport>
            ...
        </ProtocolSupport>
        <ApplicationSupport>
            ...
        </ApplicationSupport>
        <DeviceConfiguration>
            <Field Name="TimeCodeCounterfield"
            Value="0x00 0x00 0x00 0x00"/>
            <Field Name="BaseTransmitRatefield"
            Value="0x00 0x00 0xFF 0xF0"/>
            ...
        </DeviceConfiguration>
        <LinkConfiguration NoOfLinks="4">
            <FieldSubset Name="Link1">
                <Field Name="LinkStatusfield"
                Value="0xC0 0x05 0x00 0x00"/>
                <Field Name="LinkControlfield"
                Value="0x00 0x00 0x00 0x39"/>
                <Field Name="LinkDebugInformationfield"
                Value="0x00 0x00 0xA0 0x80"/>
                <Field
                Name="LinkTransmitRateRangefield"
                Value="0x00 0x7E 0x00 0x01"/>
                <Field
                Name="LinkTransmitRateDividerfield"
                Value="0x00 0x00 0x00 0x01"/>
                <Field
                Name="MinimumWatchdogDividerfield"
                Value="0x00 0x00 0x00 0x00"/>
                <Field
                Name="MaximumWatchdogDividerfield"
                Value="0x00 0x00 0x00 0x00"/>
                <Field
                Name="LinkWatchdogRateDividerfield"
                Value="0x00 0x00 0x00 0x00"/>
            </FieldSubset>
            <FieldSubset Name="Link2">
```

```
<Node Name="node_6" Label="N6" NoOfPorts="1"
LogicalAddress="0xfe" NDCP_Device="true"
NDCP_Version="2" ControlDevice="false">
    <Ports>
        <Port Number="1" Type="SpFi" TransmitRate="200"
        Connected="true"/>
    </Ports>
    <NDCP2>
        <DeviceInformation>
            <DeviceIdentification>
                <Field Name="DeviceVendorandProductID"
                Value="0x00 0x00 0x00 0x00"/>
                <Field Name="Version" Value="0x00 0x00
                0x00 0x02"/>
                <Field Name="DeviceStatus" Value="0x00
                0x00 0x00 0x00"/>
                <Field Name="RunningPorts" Value="0x00
                0x00 0x00 0x02"/>
                <Field Name="PortInformation"
                Value="0xFD 0x41 0x01 0x21"/>
                <Field Name="OwnerAddress" Value="0x00
                0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
                0x01 0x08 0x05"/>
                <Field Name="DeviceID" Value="0x00 0x00
                0x00 0x6B"/>
                <Field Name="UnitVendorAndProductID"
                Value="0x00 0x00 0x00 0x00"/>
                <Field Name="UnitSerialNumber"
                Value="0x00 0x00 0x00 0x00"/>
            </DeviceIdentification>
            <VendorProductStrings>
                <Field Name="VendorStringLength"
                Value="0x00 0x00 0x00 0x04"/>
                <Field Name="VendorString"
                Value="ITTI"/>
                <Field Name="ProductStringLength"
                Value="0x00 0x00 0x00 0x0F"/>
                <Field Name="ProductString"
                Value="Emulator on Arm"/>
            </VendorProductStrings>
            <ProtocolSupport>
                ...
            </ProtocolSupport>
            <ApplicationSupport>
                ...
            </ApplicationSupport>
        </DeviceInformation>
        <DeviceConfiguration>
            <Field Name="DeviceType" Value="0x00 0x00
            0x02 0x00"/>
            <Field Name="RoutingSwitchStatus"
            Value="0x00 0x00 0x00 0x00"/>
            ...
        </DeviceConfiguration>
        <PortConfiguration NoOfPorts="1">
            <FieldSubset Name="Port1"
            NoOfVirtualChannels="2" NoOfLanes="1">
                <Field Name="PortType" Value="0x00 0x00
                0x02 0x00"/>
                <Field Name="NetworkDiscovery"
                Value="0x80 0x00 0x00 0x00"/>
                <Field Name="PortStatus" Value="0x00
                0x04 0x02 0x01"/>
                <Field
                Name="Numberoferrorrecoveryattempts"
                Value="0x00 0x00 0x00 0x00"/>
                <Field Name="BandwidthCreditLimit"
                Value="0x00 0x00 0xCC 0xCC"/>
                <Field Name="PortControl" Value="0x00
                0x00 0x00 0x32"/>
                <FieldSubset Name="VirtualChannel0">
                    <Field Name="NodeEndPointStatus"
                    Value="0x00 0x00 0x00 0x00"/>
                    <Field Name="VirtualChannelStatus"
                    Value="0x00 0x00 0x00 0x05"/>
                    <Field Name="VirtualNetworkNumber"
                    Value="0x00 0x00 0x00 0x00"/>
                    <Field
                    Name="VirtualChannelConfiguration"
```

Fig. 1. XML representation (fragments) of a SpaceWire SpW-NDCP node (left) and a SpaceFibre NDCP v.2 node (right)

The functionality of the SoC node includes both the SpaceFibre port interface and the NDCP v.2 peripheral device, i.e. responding to NDCP requests from the network manager. The hardware selected for implementing the nodes is based on the Xilinx Zynq SoC [16]. The Zynq combines two main parts in a single chip: the Programmable Logic – an FPGA equivalent to Xilinx Artix-7 or Kintex-7 series (depending on the Zynq model), and the Processing System – an ARM Cortex-A9 processor based on the ARMv7 A architecture.

The complete node implementation includes a development board with the Zynq chip and supporting electronic elements and interface ports, together with a power supply. The specific boards used were development systems available from Trenz, with Zynq models 7030, 7035, and 7045. They are constructed as sets comprising of a system-on-a-module (SoM) board with the Zynq chip and a carrier board with all connectors for peripherals and power connectors. The essential network connector used on the carrier board is one of the eight SFP+ sockets, which are in turn connected to the gigabit transceivers on the Zynq (see Fig. 2).

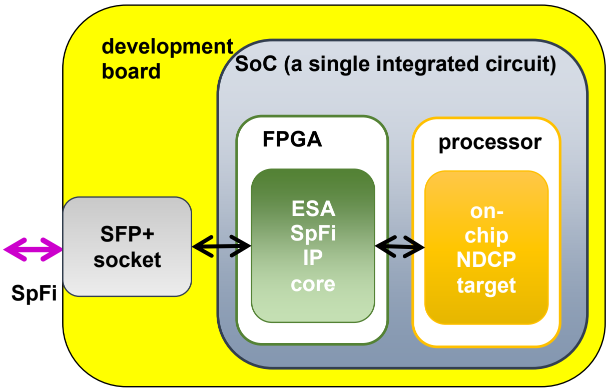


Fig. 2. SoC-based SpaceFibre node with NDCP v.2 support

The main functionality of a SpaceFibre port is provided by the ESA SpaceFibre IP core [17]. Communication between the SpaceFibre port and the SpaceFibre link uses the Zynq GTX transceivers at the lowest level and this needs a supplemental IP core. Communication between the port and the processing system uses the AXI interface on the Zynq, which also needs supplemental IP cores. Finally, there is another IP core needed for setting the configuration of the SpaceFibre port from the NDCP implementation. Thus, the following IP cores complement the ESA SpaceFibre IP core:

- an IP core for gigabit transceiver support on the Zynq,
- an IP core for transferring SpaceFibre virtual channel data between the ESA IP core and the AXI interface,
- an IP core for interfacing between AXI stream protocol (AXIS) and the DMA,
- an IP core for handling SpaceFibre configuration registers.

Some of those supplemental IP cores have been developed by ITTI in the *FiMan* project, using the VHDL language and the Xilinx Vivado development system. Others are available from Xilinx via the Vivado system. Fig. 12 shows the relations between the IP cores used. The colours, as explained in the legend, indicate the source of each of the IP cores.

The functionality of the NDCP v.2 peripheral device is provided by an NDCP software implementation on the Processing System part of the Zynq SoC as C++ based code run directly on the ARM processor. This code is responsible for managing the NDCP field base in the device and for replying to NDCP requests, either by sending back the current values of the NDCP fields or by modifying them. The NDCP fields are mapped onto the configuration registers available in the SpaceFibre IP core and the supplemental IP

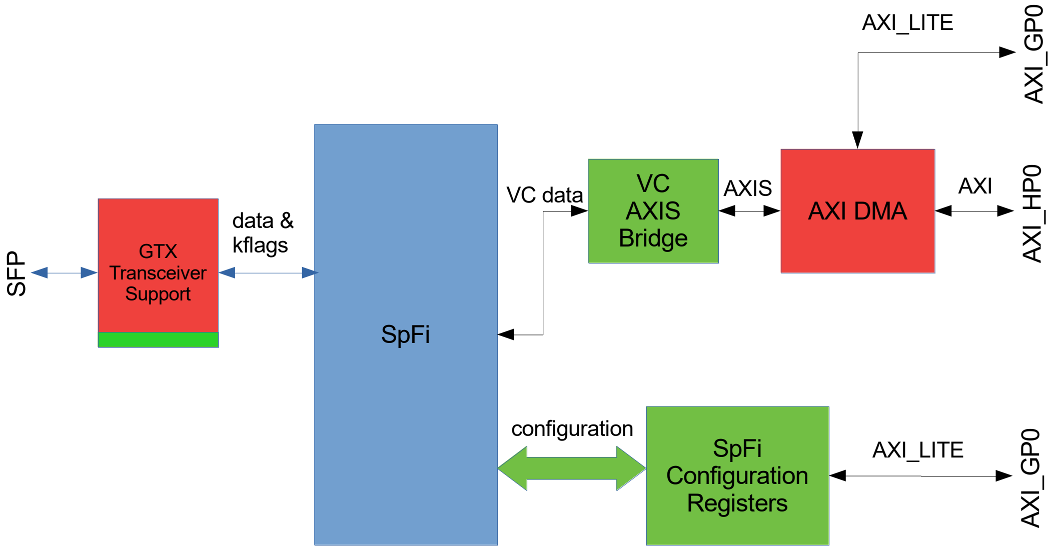


Fig. 3. IP cores used on the SpaceFibre NDCP node

Colour legend: blue: ESA IP core
red: Xilinx IP core
green: ITTI IP core
red+green: generated by ITTI based on Xilinx templates

cores perform the reading or writing the registers as appropriate, so that the values of the NDCP fields reflect the state of the SpaceFibre interface and changes of the values have effect on its configuration.

As an intermediate development product, an alternative version of the SoC NDCP SpaceFibre node with the NDCP functionality implemented on an external PC-type computer was also produced. In that case, the computer was connected to the Gigabit Ethernet port of the development board. The Processing System on the Zynq chip hosted a SpaceFibre-Ethernet gateway code that passed data between the IP cores on the Programmable Logic part of the Zynq and the PC, while the PC ran the emulator code providing the actual NDCP functionality.

A diagram of an example network prepared for the demonstration is shown in Fig. 4, and the photograph of this network – in Fig. 5. This network includes some of the SoC NDCP SpaceFibre nodes developed in the project, as well as other SpaceFibre and SpaceWire nodes and switches. Such a hybrid network, with particular devices supporting different protocols that can be used for management: SpW-NDCP, NDCP v.2, and RMAP, was successfully discovered and configured by the SPACEMAN network management tool.

VI. Conclusion

The NDCP protocol originally designed for managing SpaceWire networks has been extended to the domain of SpaceFibre and mixed SpaceWire/SpaceFibre networks. The implementation has been done in software and in

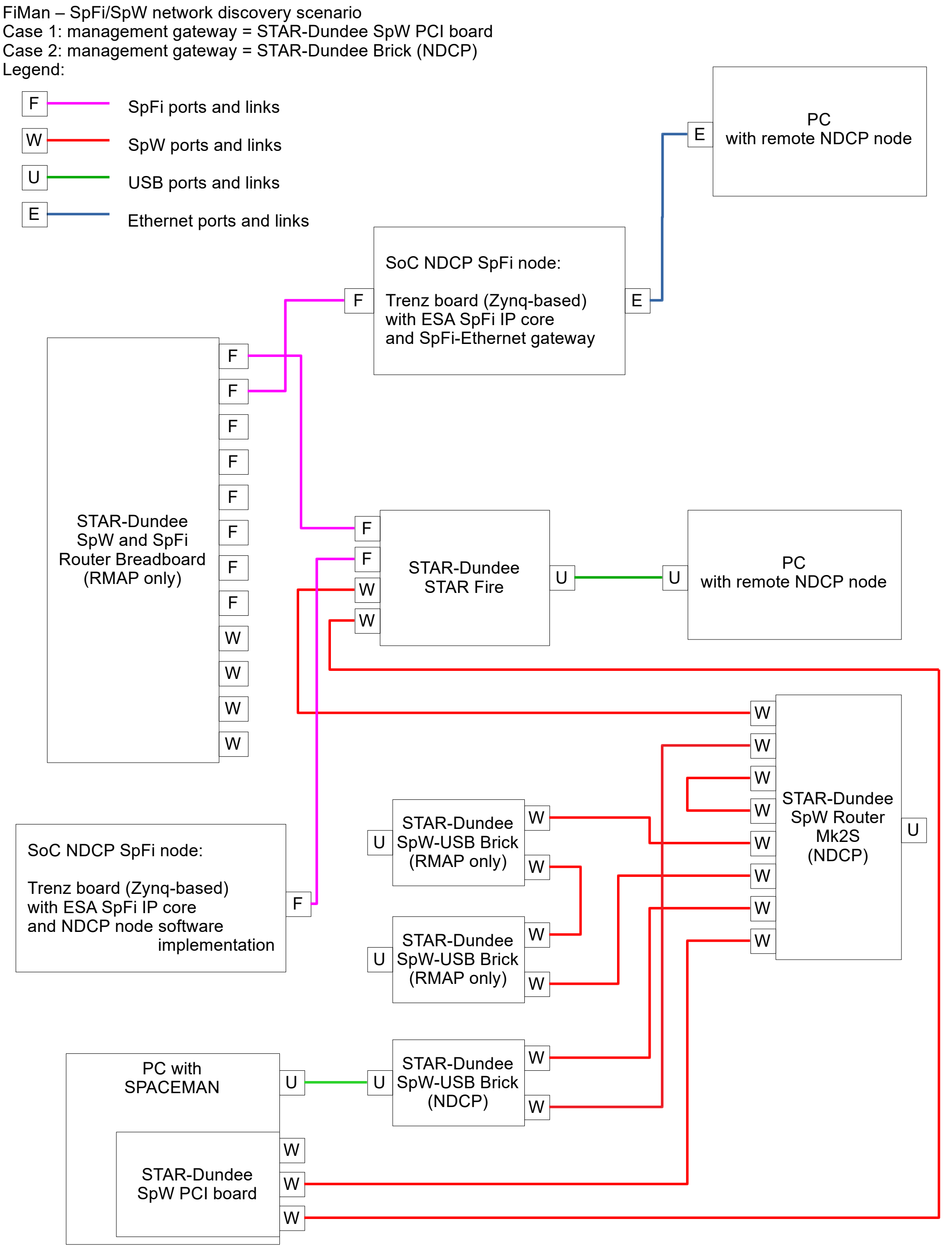

FiMan – SpFi/SpW network discovery scenario
Case 1: management gateway = STAR-Dundee SpW PCI board
Case 2: management gateway = STAR-Dundee Brick (NDCP)
Legend:
F SpFi ports and links
W SpW ports and links
U USB ports and links
E Ethernet ports and links
PC with remote NDCP node
SoC NDCP SpFi node: Trenz board (Zynq-based) with ESA SpFi IP core and SpFi-Ethernet gateway
STAR-Dundee SpW and SpFi Router Breadboard (RMAP only)
STAR-Dundee STAR Fire
PC with remote NDCP node
STAR-Dundee SpW Router Mk2S (NDCP)
SoC NDCP SpFi node: Trenz board (Zynq-based) with ESA SpFi IP core and NDCP node software implementation
STAR-Dundee SpW-USB Brick (RMAP only)
STAR-Dundee SpW-USB Brick (RMAP only)
STAR-Dundee SpW-USB Brick (NDCP)
PC with SPACEMAN
STAR-Dundee SpW PCI board
F
W
U
E

Fig. 4. Topology of the demonstrator network

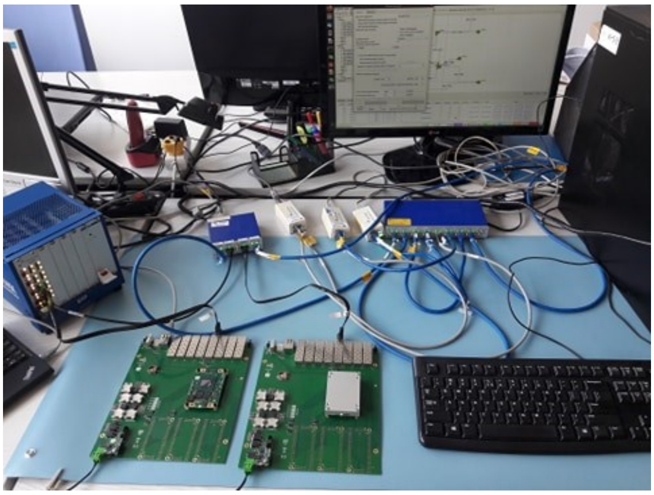


Fig. 5. Demonstrator network

FPGA/SoC. In a related activity ITTI has also developed a Zynq-based SpaceWire node based on the ESA SpaceWire/RMAP IP core, adding NDCP v.2 support similar to the SpaceFibre node described in this paper. The advantage of adopting a standard layout and addressing of configuration parameters across different technologies is device independence and interoperability of tools. Introducing version numbering for the protocol and for the individual field set structures allows different generations of devices in the same network. However, some of the management parameter definitions specified in the SpaceWire and SpaceFibre standards need refinement or clarifications as to the types, units, ranges, or default values in order to be interpreted unambiguously.

References

[1] European Cooperation for Space Standardization, Space Engineering – SpaceWire – Links, nodes, routers and networks, ECSS-E-ST-50-12C. Noordwijk: ECSS Secretariat, 2008.

[2] European Cooperation for Space Standardization, Space Engineering – SpaceWire – Links, nodes, routers and networks, ECSS-E-ST-50-12C Rev.1. Noordwijk: ECSS Secretariat, 2019.

[3] European Cooperation for Space Standardization, Space Engineering – SpaceFibre – Very high-speed serial link, ECSS-E-ST-50-11C. Noordwijk: ECSS Secretariat, 2019.

[4] European Cooperation for Space Standardization, Space Engineering – SpaceWire – Remote memory access protocol, ECSS-E-ST-50-52C. Noordwijk: ECSS Secretariat, 2010.

[5] F. Siegle and A. Leoni, "Standardization efforts for a network management and discovery protocol for SpaceFibre," Proc. 8th Int. SpaceWire Conf. Long Beach 2018, pp. 133-137.

[6] P. Nannipieri, G. Dinelli, L. Dello Sterpaio, A. Marino, and L. Fanucci, Next-Generation High-Speed Satellite Interconnect. Cham: Springer, 2021.

[7] D. Jameux, "Towards SpaceWire Plug-and-Play ECSS standard," Proc. 4th Int. SpaceWire Conf. San Antonio 2011, pp. 33-40.

[8] European Cooperation for Space Standardization, Space Engineering – SpaceWire Network Discovery & Configuration Protocol, ECSS-E-ST-50-54 Draft 1.8. Noordwijk: ECSS Secretariat, 2016.

[9] S. Fowell, "Network discovery protocols – final presentation", presentation at the TEC-ED and TEC-SW Final Presentation Days, Noordwijk, 2014.

[10] A. Tavoularis, V. Vlagkoulis, F. Kostopoulos, B. Dellandrea, T. Le Ngoc, L. Fossati, J. Ilstad, and D. Jameux, "An IP core for the SpW family of protocols," Proc. 7th Int. SpaceWire Conf. Yokohama 2016, pp. 273-280.

[11] GR718B Radiation-Tolerant 18x SpaceWire Router. 2020 Data Sheet and User's Manual, GR718B-DS-UM version 3.5, Cobham Gaisler, 2020.

[12] GR740 Quad Core LEON SPARC V8 Processor. User Manual and Data Sheet, GR740-UM-DS version 2.5, Cobham Gaisler, 2021.

[13] W. Hołubowicz, P. Lancmański, K. Romanowski, V. D. Kollias, and N. Pogkas, "SPACEMAN: A SpaceWire network management tool," Proc. 6th Int. SpaceWire Conf. Athens 2014, pp. 99-102.

[14] K. Romanowski, P. Tyczka, W. Hołubowicz, R. Renk, V. D. Kollias, N. Pogkas, and D. Jameux, "SpaceWire network management using Network Discovery and Configuration Protocol," Proc. 7th Int. SpaceWire Conf. Yokohama 2016, pp. 45-50.

[15] European Cooperation for Space Standardization, Space Engineering – SpaceWire protocol identification, ECSS-E-ST-50-51C. Noordwijk: ECSS Secretariat, 2010.

[16] Zynq-7000 SoC Data Sheet: Overview, DS190 (v1.11.1), Xilinx, 2018.

[17] SpaceFibre Port IP Core - Datasheet and User's Manual. Doc. No SPFI-DSUM-0001, Issue 1.1, Cobham Gaisler, 2016.

STP-ISS Assessment in Deterministic SpaceWire Network with MOSTNS3 SpaceWire Simulator

Barthélémy Attanasio
Thales Alenia Space
5 Allée des Gabians, 06150 Cannes La Bocca Cedex, FRANCE
barthelemy.attanasio@thalesaleniaspace.com

Brice Dellandrea
Thales Alenia Space
5 Allée des Gabians, 06150 Cannes La Bocca Cedex, FRANCE
brice.dellandrea@thalesaleniaspace.com

Elisa Ballatore
Thales Alenia Space
5 Allée des Gabians, 06150 Cannes La Bocca Cedex, FRANCE
elisa.ballatore@thalesaleniaspace.com

David Jameux
European Space Agency
2200 AG Noordwijk, THE NETHERLANDS
david.jameux@esa.int

***Abstract*— SpaceWire is a protocol developed for space use and specifically for spacecraft. Its numerous advantages enable a wide use in the space industry in particular comparing to the well-known MIL-STD 1553 bus because SpaceWire combines simple, low-cost implementation, with high performance and architectural flexibility. In fact it is a low consumption and high data rates communication link which can even be used for a network. SpaceWire networks are becoming more complex and some problems can occur in different configuration which lead to a need of protocol improvement.**

ISS-Reshetnev has developed a specific standard based on SpaceWire which brings some mechanisms for providing some Quality of Service (QoS). Using these QoS in a network can bring a high level of determinism on the spacecraft for good equipment and behavior management.

This paper aims at exploring and comment on the STP-ISS (Streaming Transport Protocol Information Satellite System) by implementing this additional SpaceWire layer on a NS3 network simulator.

Keywords—SpaceWire, STP-ISS transport protocol, MOSTNS3 Simulator, ESA, Thales Alenia Space, ISS-Reshetnev

I. Introduction (*Heading 1*)

Nowadays there is a number of transport protocols intended to operate over SpaceWire. They are: RMAP (Remote Memory Access Protocol), CPTP (CCSDS Packet Transfer Protocol), STUP (Serial Transfer Universal Protocol), JRDDP (Joint Architecture Standard Reliable Data Delivery), and STP. Each of them is designed to solve its particular tasks. However, there is no SpaceWire oriented transport protocol providing reliability, guaranteed services and scheduling.

Other standard based on SpaceWire offers a deterministic layer like the STP-ISS transport protocol; this is the case of SpaceWire-D and SpaceWire-NMS.

II. STP-ISS Transport Protocol Presentation

A. Overview of the STP-ISS transport protocol

STP-ISS is a transport layer protocol which operates over the basic SpaceWire protocol. It has been developed by the Saint-Petersburg State University of Aerospace Instrumentation and the JSC "Academician M.F. Reshetnev Information Satellite Systems". STP-ISS provides data transmission between remote network nodes with the required quality of service in accordance with data flow priorities. This protocol gives data resending ability in case of error detection in the received data, thus ensuring the reliability of data delivery.

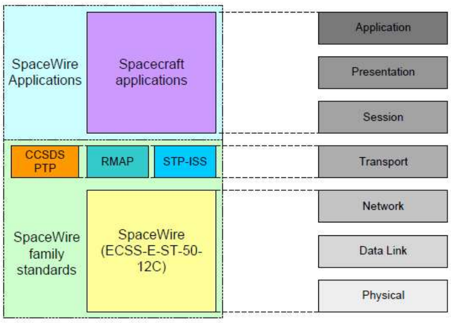


Fig. 1. STP-ISS in the SpaceWire network architecture

The STP-ISS protocol defines different interfaces to interact with other layers. This operates to control the link between the network and the application layer.

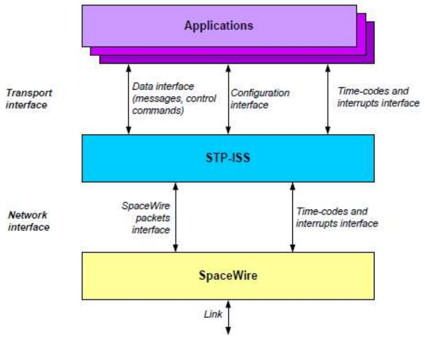

Applications
Transport interface
Data interface (messages, control commands)
Configuration interface
Time-codes and interrupts interface
STP-ISS
Network interface
SpaceWire packets interface
Time-codes and interrupts interface
SpaceWire
Link

Fig. 2. STP-ISS Transport Protocol Interfaces

Three links represent the transport interface:

- Data interface: for messages (Control Command, Urgent message, Regular message..)

- Configuration interface
- Timecodes and interrupts interface

Two links represent the network interface:

- SpaceWire packets interface: for the transmission of application messages
- TimeCodes

In order to differentiate the STP-ISS protocol from other data transport protocols, the PID (Protocol Identifier) field, defined in the standard ECSS-E-ST-50-51C, take the specific value 252.

Concerning the data sent by this transport protocol on the network, there is no change comparing to SpaceWire cargo. Indeed, this deterministic layer is at transport level, which explains why the SpaceWire cargo sent is not modified.

B. Determinism providing & Quality of Service

The main service brought by STP-ISS is an additional level of determinism by providing different Quality of Service thanks to a dedicated header. STP-ISS transmits user messages according to four different Quality of Service (QoS):

- Priority QoS : This QoS defines the priority for each data type; data with the higher priority will be transmitted first. It is the main QoS which is implemented in any case. However, there are 9 priority levels, so 9 types of buffer, are they really all necessary in space application?
- Best effort QoS : This effort QoS provides data transmission through the SpaceWire network without any service except that the receiver shall then check the correctness of the data and transmit the packet to the application layer with the error indication flag.

Priority QoS and Best Effort QoS are the basic use of STP-ISS, but they are optional because additional and more interesting QoS could replace it:

- Guaranteed QoS : This one acknowledges the correct data delivery by the transmission of the acknowledgement. Moreover, it provides the possibility of send back packets at the transmitter level in case no acknowledgement is received after timeout of the resent timer
- Scheduling QoS: This QoS provides data transmission in accordance with a schedule which is defined for the given node during protocol configuration. This adds a feature which is required for some cases where a time minimum bandwidth is reserved for one end-point.

Depending on the level of determinism to reach, these two last QoS can be activated simultaneously.

III. Organization and Work Logic

The purpose of this paper is to work on the STP-ISS specification and to implement this additional SpaceWire layer over a Thales Alenia Space simulator. Thanks to ESA and Saint-Petersburg University collaboration, the STP-ISS specification (STP-ISS-14E rev2) was available in English ready to be used for investigation. This paper will present some comments around the different mechanisms detailed in the STP-ISS specification.

IV. MOSTNS3 Simulator Presentation

Embedded network modeling is important during all phases of a system design project. It allows to make decisive choices regarding the nodes' architecture, and the network type to be used.

To meet this specific need, Thales Alenia Space has launched the MOSTNS3 project for Modeling of On-Board Spacecratf Traffic. It allows the study of the behavior of a network when traffic is present. This simulator is based on NS3 (network simulation 3), a discrete event simulation software. The simulator offers the user a graphical interface called MOST-GUI, in which the user can create the topology of his network by adding generic nodes and routers or specific nodes and routers. Moreover, the user can define the traffic sent over the network. Various parameters can be modified on the nodes: the speed, the automatic activation of the ports, and the transmission interval between each Nchar. The connections and the generation of packets are also configurable: the user can define the start time of the generation, the period, and the value of some field of the header. Finally, many measurements such as buffer occupancy, packet latency, type of data sent and received, or state machine can be carried out.

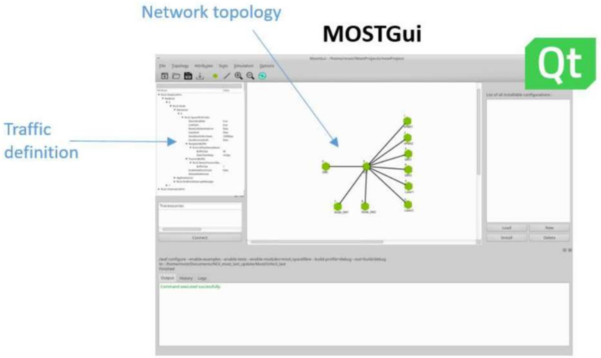


Fig. 3. MOSTGUI Presentation

The **Erreur ! Source du renvoi introuvable.** presents MOSTGUI environment in which the user defines the totality of the network: construction of the architecture and definition of the traffic.

V. Development

A. A new end-point in MOSTNS3 Simulator: STP-ISS node

The STP-ISS protocol has been developed in the model brick of the MOSTNS3 simulator. This brick contains all the functions detailed in the standard. In the helpers brick of MOSTNS3, all the function are assembled in order to create a STP-ISS end-point. **Erreur ! Source du renvoi introuvable.** represents the integration of this new data transport protocol in MOST:

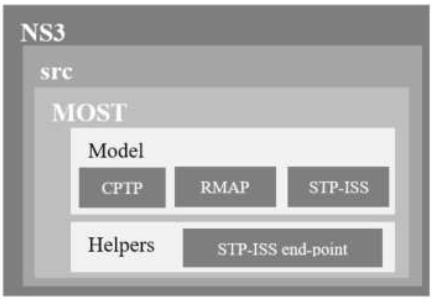


Fig. 4. Representation of the STP-ISS brick in MOSTNS3

Development has an impact at two physical levels: at the level of the sender and the level of the receiver. By breaking down the transmission of a packet according to the different network layers, it is the application layer that has been mainly impacted with the introduction of a transport layer, as shown in the **Erreur ! Source du renvoi introuvable.**:

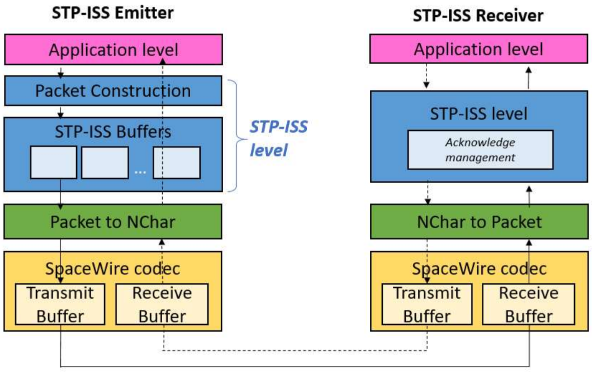


Fig. 5. Transmission architecture of a STP-ISS data over SpaceWire network

The construction of the different types of STP-ISS data as well as the management of the different QoS had to be implemented in MOSTNS3 in order to use this data transport protocol on the SpaceWire network, which were already implemented in MOSTNS3.

B. Remarks

During the development phase and test phase, some problems or drawbacks in the specifications were noted and are explained hereafter.

The first is the lack of information on QoS management at a router level because STP-ISS is only operating at endpoint level. In a case where two messages of different priority would arrive simultaneously on a switch, the latter uses wormhole routing, therefore arbitration follows a round-robin as defined in the SpaceWire standard. This round-robin can be defined in different ways. Actually, the two ways used by switches are random choice, or ascending / descending order. Therefore the messages will be transmitted according to the index of their input port in the router and not by their own priority or any STP-ISS information that the message could own before it is sent to the network.

The second problem noted is on the Scheduling QoS. Indeed, there is the existence of a lifetime timer which enables to delete the message if it is not sent to the network. The problem occurs if the lifetime timer is less than the duration of a time-slot. In this case, if the packet is stored in the buffer during a time-slot because the packet is not allocated this time slot, it will be directly deleted from the buffer because of its lifetime timer and never being sent.

VI. Tests and Results

A. MOSTNS3 Simulation with a SpaceWire router

The objective is to present the tests that have been done, highlighting the possible problems identified in the specification of STP-ISS-14E.

To highlight the problem concerning the lack of information on priority routing, the topology in **Erreur ! Source du renvoi introuvable.** has been used. The network is composed of four STP-ISS end-points and one SpaceWire router.

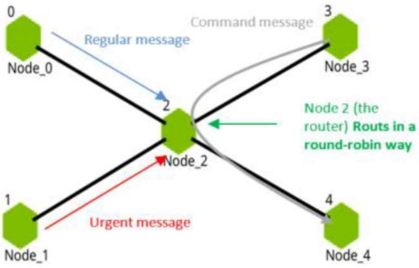


Fig. 6. Network architecture of the test modeled with MOSTGUI

Traffic definition :

- Node 0 : Regular Message to node 4
- Node 1: Urgent Message to node 4
- Node 3: Control Command to node 4

If the router used the priority to route the messages and not the round-robin, the node 4 should receive the messages in order of priority as in the **Erreur ! Source du renvoi introuvable.**:

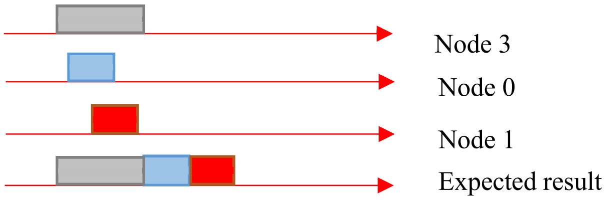
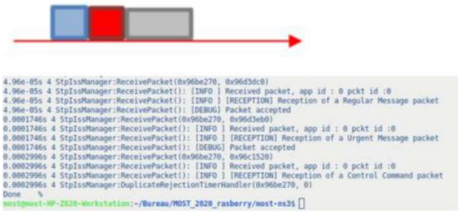


Fig. 7. Expected results of the priority arbitration

As this uses the round-robin arbitration, messages are received by node 4 as follows:



```
4.96e-05s 4 StpIssManager:ReceivePacket(0x96be270, 0x96d3dc0)
4.96e-05s 4 StpIssManager:ReceivePacket(): [INFO ] Received packet, app id : 0 pckt id :0
4.96e-05s 4 StpIssManager:ReceivePacket(): [INFO ] [RECEPTION] Reception of a Regular Message packet
4.96e-05s 4 StpIssManager:ReceivePacket(): [DEBUG] Packet accepted
0.0001746s 4 StpIssManager:ReceivePacket(0x96be270, 0x96d3eb0)
0.0001746s 4 StpIssManager:ReceivePacket(): [INFO ] Received packet, app id : 0 pckt id :0
0.0001746s 4 StpIssManager:ReceivePacket(): [INFO ] [RECEPTION] Reception of a Urgent Message packet
0.0001746s 4 StpIssManager:ReceivePacket(): [DEBUG] Packet accepted
0.0002996s 4 StpIssManager:ReceivePacket(0x96be270, 0x96c1520)
0.0002996s 4 StpIssManager:ReceivePacket(): [INFO ] Received packet, app id : 0 pckt id :0
0.0002996s 4 StpIssManager:ReceivePacket(): [INFO ] [RECEPTION] Reception of a Control Command packet
0.0002996s 4 StpIssManager:DuplicateRejectionTimerHandler(0x96be270, 0)
Done    %
most@most-HP-Z820-Workstation:~/Bureau/MOST_2020_rasberry/most-ns3$
```

Fig. 8. Simulation results using MOSTNS3

B. *Advantages and disadvantages of STP-ISS transport protocol*

The different QoS bring significant determinism to the SpaceWire networks. Indeed, Priority QoS allows high importance packets to be transmitted before less important packets, but the problem with this QoS is that some low priority packet will not be sent if high priority packets are still being generated. This is where Scheduling QoS brings real added value: one or more time-slot can be allocated to each type of STP-ISS buffer which will allow low priority data to be sent anyway. In addition, Guaranteed QoS makes it possible to know the packet's state and guarantee its correct transmission and reception.

This determinism is interesting at the STP-ISS end-points level, but in slightly more complex network architecture with SpaceWire routers, the determinism is lost.

C. *Proposed solution: STP-ISS layer in a SpaceWire router*

To solve this problem at the router side, it can be proposed to implement a mechanism which deals with the priority inside the switch by reading the STP-ISS header or at least to filter the bandwidth at the switch level with local knowledge of the epoch authorizations. This would buffer with the same number of buffers as the STP-ISS standard. It would be a new kind of selection without a round-robin but following the data flow rules introduced by the STP-ISS protocol with the corresponding priorities.

This takes the hypothesis of modifying the switches by implementing several buffers to manage the priorities. It has an impact at Hardware and Firmware levels with additional buffers and registers.

At the level of the router, it is possible to know the type of packet received and, therefore, its priority, the mechanism which checks the priority and manages it , does not exist.

VII. Conclusion

The STP-ISS SpaceWire layer is an attractive solution to improve the quality of the SpaceWire protocol. The priority mechanism enables to configure of hierarchical traffic. Beyond this basic feature, the scheduling and guaranteed QoS bring a high level of determinism having guaranteed delivery and guaranteed bandwidth which are important features. However, as it has been explained during the study, there is a lack of deterministic mechanism implemented at the router level, for example, it is not possible to prioritize the different equipment at the switch level.

This paper was focused only on the simulation part of the mechanism of the STP-ISS data transport protocol. There was no HW implementation and, therefore, no FPGA footprint size estimation. This will depend on the size of the buffers as well as the number of buffers linked to the traffic description.

References

[1] ESA (European Space Agency). Standard ECSS-E-50-12CRev1, “Space engineering. SpaceWire – Links, nodes, routers and net-works. European cooperation for space standardization”. Noordwijk: ESA Publications Division ESTEC, 2019. 129 p.

[2] Yuriy Sheynin, Irina Lavrovskaya, Valentin Olenev, Ilya Korobkov, Dmitry Dymov, Sergey Kochura: “STP-ISS Transport Protocol for Spacecraft On-board Networks”. SpaceWire networks and protocols, Long Paper. Proceedings of 6th International Conference SpaceWire 2014 Program, Athens, Greec, 2014.pp.26-31.

[3] Yuriy Sheynin, Valentin Olenev, Irina Lavrovskaya, Ilya Korobkov, Sergey Kochura, Sergey Openko, Dmitry Dymov: “Second Revision of the STP-ISS Transport Protocol for On-Board SpaceWire Networks”. PROCEEDING OF THE 17TH CONFERENCE OF FRUCT ASSOCIATION.

[4] Valentin Olenev, Irina Lavrovskaya, Nadezhda Chumakova: “Software-to-Hardware Tester for the STP-ISS transport protocol verification”, proceedings of 2016 International SpaceWire Conference. Yokohama, Japan 2016.

[5] Steve Parkes, Albert Ferrer Florit: “SpaceWire-D Deterministic Control and Data Delivery Over SpaceWire Networks”, University of Dundee Dundee, DD1 4HN Scotland, UK.

[6] Barthélémy Attanasio, Brice Dellandrea, HJ.Boeestermoeller: “SpaceWire Management Service Suite, D1 – Requirements Consolidation Report”.

[7] ESA (European Space Agency) Standard ECSS-E-50-51C “ Space engineering. SpaceWire – SpaceWire Protocol Identification”, Noordwijk: ESA Publications Division ESTEC, 2008. 15 p.

Test & Verification (Long)

Simulation and hardware validation of SerDes links for SpaceFibre

1st Gabriela Mystkowska
Warsaw University of Technology
Warsaw, Poland
gabriela.mystkowska.stud@pw.edu.pl

2nd Felix Siegle
European Space Agency
Noordwijk, Netherlands
felix.siegle@esa.int

3rd David Steenari
European Space Agency
Noordwijk, Netherlands
david.steenari@esa.int

***Abstract*—The advancement of complex on-board data-handling networks constitutes a need for high-speed data links. SpaceFibre has been developed with its quality of service (QoS), fault detection, isolation and recovery (FDIR) capabilities and high data rates (up to 6.25 Gbps per lane). In 2019 a common ECSS standard has been published, which describes the basis of the protocol and its physical layer. However, a compliance testing methodology has not been described yet. Thus, this paper intends to provide a set of hardware validation and simulation-based verification strategies for SerDes links that result in best practice design guidelines.**

The methodologies have been evaluated with simulations and hardware tests of a FPGA-based data processing board, supporting the SpaceFibre standard on both the backplane and the front-panel connectors.

***Index Terms*—SpaceFibre, SerDes, compliance, testing, simulation, validation**

I. Introduction

SpaceFibre (SpFi) is a high-speed serial link and network technology for on-board spacecraft use. It is a successor of SpaceWire with improved data rate by a factor of 10 (up to 6.25 Gbps per line and over 20 Gbps in multi-lane configuration), reduced cable mass and fitted with galvanic isolation. SpFi supports both fiber-optic and electrical cables. It provides coherent quality of service (QoS) and improved fault detection, isolation and recovery (FDIR) capabilities compared to SpaceWire. The SpFi signal is a DC free NRZ with 8b/10b coding. [6]

Advanced Data Handling Architecture (ADHA) is a new approach for designing and producing spacecraft subsystems. Its main focus is higher level of integration of On-Board Computer (OBC), Solid State Mass Memory (SSMM), Remote Interface Unit (RIU), Global Navigation Satellite System (GNSS): to reduce the mass, size and power of the Data Handling System (DHS) equipment, limit the number of interfaces, the harness and consequently the assembly, integration and tests (AIT) effort. It is achieved by implementing a modularity concept to support the interchangeability and the interoperability between different missions. ADHA aims to improve performance of Data Handling System (DHS) by using multicore processors and new generations of high speed networks and links (e.g., SpFi). It promotes the use of COTS components. [7] [11] [9] The modular approach of ADHA constitutes a need for a common compliance testing technique in order to validate individual systems. However, the ECSS standard [6] only defines the eye mask for IC input/output, not for the entire unit.

Two different compliance testing methodologies of commercial high-speed protocols that use NRZ signal with 8b/10b coding (USB 3.0 and DisplayPort 1.0) were analysed. [3] [1] The USB 3.0 Electrical Compliance Methodology describes the compliance values at the end of a compliance channel (including PCB routing, connectors and cables) without characterization of additional fixtures before testing. It specifies different patterns for testing different properties, e.g., sequence of 1 and 0 at max switching rate (Nyquist freq.) to eliminate Deterministic jitter (Dj) or only scrambled logical idle signal for Rx testing. Different values for Random jitter (Rj), Deterministic jitter (Dj) and Total jitter (Tj) for both Transmitter (Tx) and Receiver (Rx) are specified. For Tx testing, CTLE equalizer is simulated by the measuring equipment. [5]

The VESA DisplayPort PHY Compliance Test Standard describes the test set-up with additional fixtures and requires its characterization before performing measurements. It specifies different patterns for every test, some criteria are tested under only one pattern. It only considers Tj, different pass/fail values at different test points are specified. All tests are performed with pre-emphasis on the Tx. It defines signal attenuation of Rx signal at test points, and describes noise measurements (mainly focused on the cable properties). [2]

Both compliance testing standards use the Dual Dirac Jitter Model to calculate the Tj and require at least 10^6 consecutive UI for jitter measurements.

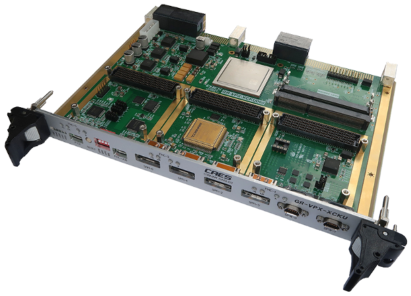


Fig. 1. High-Performance Compute Board (HPCB) from Cobham Gaisler AB[12]

In order to obtain data for this paper an High-Performance Compute Board (HPCB) from Cobham Gaisler AB with Xilinx KU060 FPGA was tested and simulated. The HPCB is presented in Figure 1. Compared to existing technology, the HPCB platform provides more computational resources on-board spacecrafts to process high bit-rate payload data before downlink, thus reducing bandwidth requirements and improving reaction times of space systems.

The target applications include on-board payload processing for optical and radar instruments, as well as visual navigation. The board can be integrated in payload data handling units, mass-memory and/or on-board computer to enable functions such as high-performance on-board image processing, machine vision and standard CCSDS 123.0 image compression. The board design is optimized for the data handling and processing of multiple instruments simultaneously. [8] [10]

HPCB provides 24 bidirectional HSSLs using front-panel connectors (4, eSATA), FMC connectors (4+4+4) and VPX backplane connectors (4+4). Only front-panel connectors were tested and simulated due to available testing hardware. Differences between PCB routing between front-panel ports are presented in Table I.

TABLE I
PCB routing for different front panel ports on HPCB board (data obtained using Mentor HyperLynx net statistics)

Port	**Track length [cm]**	**Total copper delay [ns]**	**Net capacitance [pF]**	**Resistance [Ω]**	**Impedance [Ω]**
SPFI-0	21.985	1.4012	26.204	1.584	54.1
SPFI-1	22.302	1.4269	26.308	1.635	54.6
SPFI-2	28.349	1.82	34.145	2.092	53.7
SPFI-3	29.639	1.901	34.799	2.187	54.9

II. Methods

First simulations of HSSLs Tx on HPCB were performed in order to obtain a basic idea of how the board performs. Next, Tx hardware tests were performed to compare the simulation to a real life unit. Additionally, Rx hardware tests were performed to obtain more data about the HPCB. A concept of test set-up is presented in Figure 2 and the lab test set-up is presented in Figure 3.

A. Simulation

Simulations were performed using Mentor HyperLynx software. Only the HPCB Tx without the harness (FPGA, PCB traces and connector - the elements within the dotted line in Figure 2 A) was simulated. A model of the GTH transmitter provided by Xilinx was used. A model of the connector was not used. Resistors and capacitors were simulated based on their value; a precise model was not used. An additional parallel DC 50 Ω resistor was used as termination at the connectors.

The following factors have been considered:

- Four front-panel ports, due to different routing

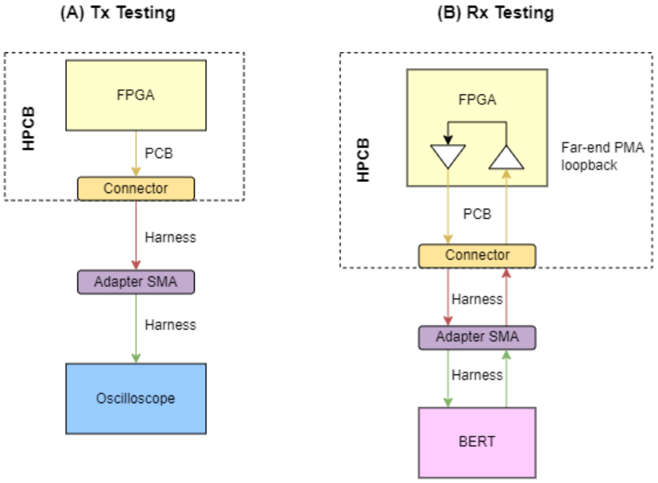


Fig. 2. Tx and Rx testing diagram

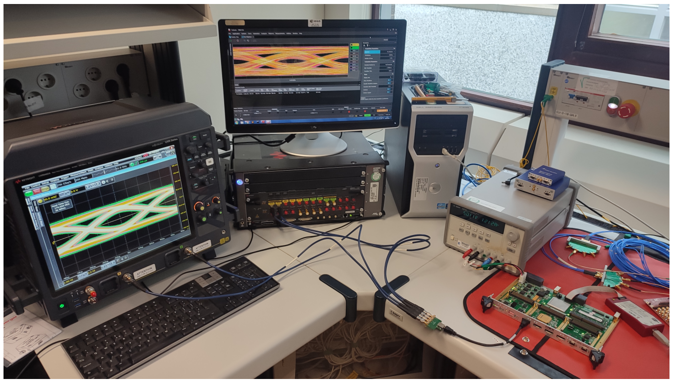


Fig. 3. Test set-up from left: Oscilloscope Keysight UXR0402A, 40 GHz, 256 GSa/s and Teledyne SDA 820Zi-B, 20 GHz, 80 GSa/s (not in Figure 3), BERT Keysight M8041A 8.5 Gb/s, Power supply Agilent E3631A, HPCB board, SpFi cables (0.25 m, 1 m, 4 m), SMA - eSATA connector swap board, SMA cable 1m

- Voltage swing: 800 mV, 950 mV and 1080 mV – the ECSS standard [6] describes that the transmitter voltage swing shall be between 800 mV and 1600 mV, therefore values below 800 mV were not simulated and since GTH maximum available voltage swing is 1080 mV, higher values were not simulated
- Data rates: 1 Gbps, 1.25 Gbps, 2 Gbps, 2.5 Gbps, 3.125 Gbps, 5 Gbps, 6.25 Gbps - according to ECSS standard [6]
- Patterns: PRBS and 8b/10b – PRBS is a commonly used pattern, available in every SerDes device and the SpFi uses 8b/10b coding
- No emphasis, no equalization – the ECSS standard [6] only recommends the use of emphasis and equalizer, but does not specify the parameters, therefore they were not considered in the simulations

B. Hardware Tests

First the HPCB was connected to itself through a cable (external loop back) in order to check weather the FPGA was programmed correctly. This test is too optimistic, because the receiver PLL is clocked by the same clock as the transmitter. However, it is a good practice technique to distinguish hardware problems from FPGA/SerDes configuration problems.

For Tx testing the HPCB was connected as presented in Figure 2 A. Different harnesses were used to determine their impact on the signal - 3 SpFi cables (0.25 m, 1 m and 4 m). Different signal settings were used: data rates (1.25 Gbps and 6.25 Gbps), voltage swings (660 mV, 840 mV, 950 mV, 1080 mV) and pre-cursor settings (4.44 dB, 6.47 dB, 8.52 dB, 12.96 dB). Four different SpFi front-panel ports were tested, each has different PCB routing as described in Table I. The FPGA design uses SerDes internal PRBS generator without 8b/10b coding.

For Rx testing there are two approaches:

- characterisation of PCB
- simulating real-case signal using a BERT

For this paper the first approach was taken, because of lack of signal simulation. The test set-up is presented in Figure 2 B. First, each of the front-panel ports were tested for the minimal voltage swing that would not cause errors. Then each port was tested for jitter tolerance factory pre-set curves (with voltage swing set to 1000 mV on the BERT):

- USB 3.0 5G (start frequency: 500 kHz, stop frequency: 50 MHz)
- SAS2 6G no SSC (start frequency: 240 kHz, stop frequency: 15 MHz)

Those two protocols were chosen because of their similarity to SpFi - data rate of respectively 5 Gbps, and 6 Gbps, 8b/10b coding. [3] [4] For all Rx test, the HPCB Tx voltage swing was set to 950 mV, without emphasis, and on the Rx side DFE was enabled (default settings for this SerDes), the SerDes was set to far end loopback PMA.

The FPGA SerDes has a built in eye scan function on the Rx side that measures eye diagrams as received by the FPGA, after equalization. This function was used for Rx testing, it gives a rough idea how well the signal is received and how well it's recovered using the equalization function.

III. Results and Discussion

A. Tx Simulation

Simulation results are presented in Figure 4 and Figure 5. An eye mask presented on the eye diagrams matches the far-end serial eye pattern mask described in the ECSS standard. [6]

With the increase of data rate, the eye height and eye width decrease as presented in Figure 6. The eye height decreases with the decrease of the voltage swing, as presented in Figure 6 The eye opening differs for PRBS-19 signal and 8b/10b signal, but the difference is insignificant as presented in Figure 6. The average values were calculated from different SpFi front-panel ports.

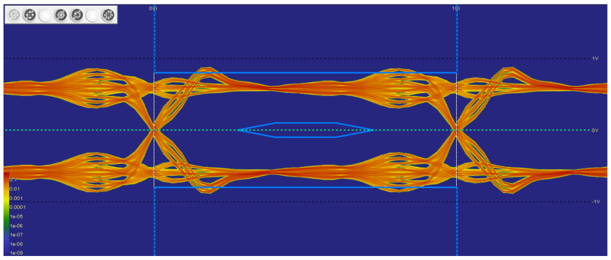


Fig. 4. Eye diagram simulation of SPFI-0 front-panel port at 1.25 Gbps, 1080 mV voltage swing. The eye height is 1007 mV

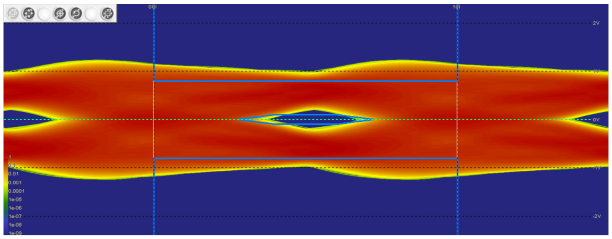


Fig. 5. Eye diagram simulation of SPFI-0 front-panel port at 6.25 Gbps, 1080 mV voltage swing. The eye height is 314 mV

B. Tx Hardware Test

For hardware tests, first the Tx was tested. The eye diagram for data rate 1.25 Gbps is presented in Figure 7 and for 6.25 Gbps - Figure 8. The eye mask matches the far-end serial eye pattern mask described in the ECSS standard. [6] The signal, in both cases, fits within the eye mask limits.

Tx measurement results for one of the ports is presented in Figure 9. For 4 m cable there's only one measurement for maximal voltage swing, because for lower values the signal was too attenuated to be measured by the oscilloscope.

The pre-cursor settings may impact the signal in a positive way as presented in Figure 10 A, but may also impact the eye diagram negatively as shown in Figure 10 B. The effect of pre-cursor settings depends strongly on the particular set-up (PCB routing, harness length and type, signal type and data rate).

C. Rx Hardware Test

The lowest voltage swing set on BERT that wouldn't cause errors was 25 mV, an eye diagram after equalisation measured by the HPCB Rx is presented in Figure 11. If there was an area with BER of $1.0e-6$ (indicated as blue on the eye diagram) the FPGA was still able to correctly receive the signal. It's also worth noticing that the signal was attenuated by the harness and SMA adapter, but the signal itself had little to no jitter since the clock source of BERT signal is precise in terms of jitter.

HPCB Rx jitter tolerance measurements are presented in Figure 12 and Figure 13. For both diagrams, the solid lines indicate the acceptable compliance ranges for the selected protocols. For both cases, the HPCB is compliant with the defined curves.

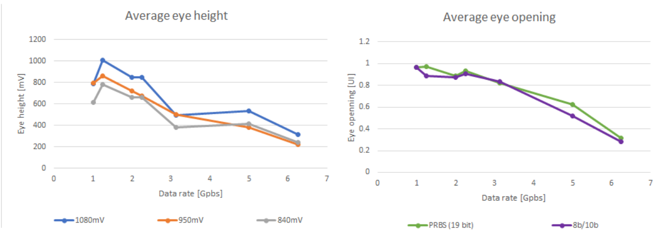


Fig. 6. Average eye height and eye width from simulation

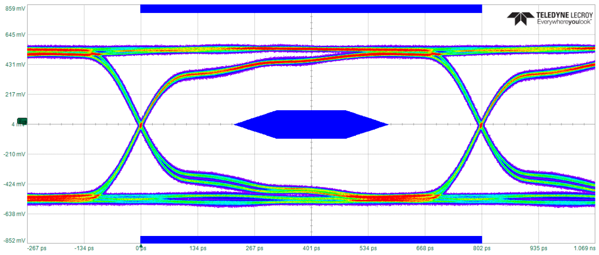


Fig. 7. Eye diagram of SPFI-0 front-panel port at 1.25 Gbps, 1080 mV voltage swing. The eye height is 782 mV

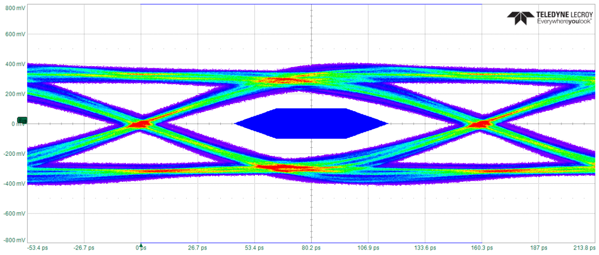


Fig. 8. Eye diagram of SPFI-0 front-panel port at 6.25 Gbps, 1080 mV voltage swing. The eye height is 383 mV

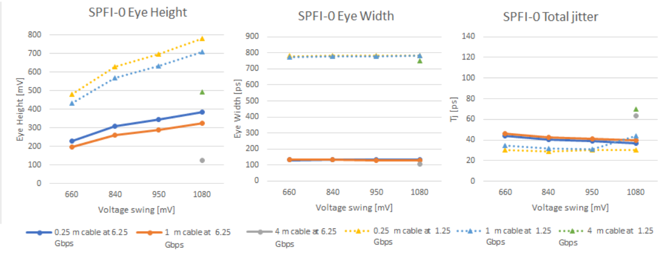


Fig. 9. Tx test results for SPFI-0 front panel

IV. Conclusions

Eye diagrams for Tx simulation and test optically look different (for 1.25 Gbps Figure 4 and Figure 7, for 6.25 Gbps Figure 5 and Figure 8). The simulations are a worst case scenario for the tested HPCB, but overall they gave a rough idea of how the board performs. In Table II the comparison of Tx eye height and width values for simulation and test is presented.

TABLE II
Comparison of eye diagram parameters in Tx simulation and test

Data Rate	Parameters	Simulation	Test
1.25 Gbps	Eye height	982 mV	696 mV
	Eye width	0.9694 UI	0.9775 UI
6.25 Gbps	Eye height	254 mV	353 mV
	Eye width	0.3125 UI	0.3375 UI

It's difficult to determine one value of recommended pre-cursor settings since it depends strongly on the particular set-up (PCB routing, harness length and type, signal type and data rate), therefore we recommend that the different pre-cursor settings were tested upon assembly and one setting was chosen based on performance in this particular system.

The Rx tests showed impressive capabilities of the SerDes, it can correctly receive signal of only 25 mV voltage swing. It also proved to be compliant with USB 3.0 and SAS2 jitter tolerance. The pre-set curves for those protocols partly cover the same jitter frequencies, but the results are different, which means the jitter component mix for this pre-set curves are different. What kind of jitter type is considered for those protocols, and most importantly how the jitter mix was determined, remains a question.

During the testing process, after analysing different compliance testing protocols for multiple standards, both high-speed and others, we were unable to determine how the numerical values of pass/fail criteria presented in the standards were determined. Answering this question is crucial for writing a SpaceFibre Compliance Test Standard in order to create a repeatable test set-up.

This paper's aim was to provide a detailed description of simulations and tests for characterising SpFi HSSL. Without a common compliance testing standard, we recommend all units using SpaceFibre should be tested in the way described in this paper, in order to maintain repeatable testing techniques.

The test equipment mentioned in this paper - oscilloscopes Keysight UXR0402A,(40 GHz, 256 GSa/s) and Teledyne SDA 820Zi-B (20 GHz, 80 GSa/s), BERT Keysight M8041A 8.5 Gb/s are available for industry in ESA ESTEC labs.

V. Acknowledgments

The research leading to these results was done as part of an internship at ESA ESTEC. I would like to thank Felix Siegle, my internship mentor, for his help, advice and guidance. I would also like to thank Malte Bargholz, Ric Dengel and TEC-EDD section for their support.

References

[1] *VESA DisplayPort Standard*. 2006.
[2] *VESA DisplayPort PHY Compliance Test Standard*. 2007.
[3] *Universal Serial Bus 3.0 Specification Universal Serial Bus 3.0 Specification, Revision 1.0*. 2008.
[4] *Serial Attached SCSI Standard*. 2009.
[5] *USB 3.0 Electrical Compliance Methodology White Paper Revision 0.5*. 2009.

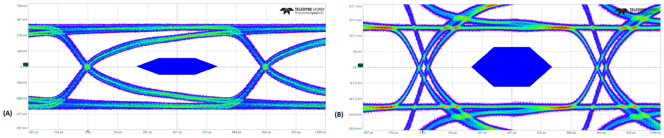


Fig. 10. Eye diagram affected by pre-cursor settings (A) 4.44 dB, (B) 8.52 dB at 1.25 Gbps, 1080 mV voltage swing

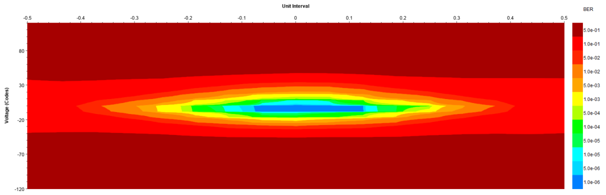


Fig. 11. Eye diagram measured by HPCB Rx after DFE equalisation for 6.25 Gbps PRBS signal with 25 mV voltage swing from BERT.

[6] *ECSS-E-ST-50-11C SpaceFibre - Very high-speed serial link*. 2019.

[7] Wahida Gasti. “Once upon a Time ....ADHA”. In: 14^{th} ESA Workshop on Avionics, Data, Control and Software Systems. 2020.

[8] Joaquín España Navarro et al. *High-Performance Compute Board - A fault-tolerance module for on-board vision processing*. 2021.

[9] Julian Bozler. *Advanced data handling architecture for EO satellites*. 2021.

[10] Joaquín España Navarro. *High-Performance Compute Board. COTS Acceleration for Earth Observation Applications*. 2021.

[11] Dario Pascucci. *Modular and Interoperable Advanced Data Handling Architecture (ADHA) for Earth Observaton (EO) Satellites*. 2021.

[12] Cobham Gaisler AB. *GR-VPX-XCKU060 Carrier Board Data Sheet & User Manual*. 2022.

Acronyms

ADHA	Advanced Data Handling Architecture
AIT	assembly, integration and tests
BER	Bit Error Rate
BERT	Bit Error Ratio Tester
COTS	Commercial Off-The-Shelf
CTLE	Continuous-Time Linear Equalizer
DFE	Decision Feedback Equalizer
DHS	Data Handling System
Dj	Deterministic jitter
ECSS	European Cooperation for Space Standardization
ESA	European Space Agency
FDIR	fault detection, isolation and recovery
FPGA	Field-Programmable Gate Array
GNSS	Global Navigation Satellite System
HPCB	High-Performance Compute Board
HSSL	High Speed Serial Links
IC	integrated cirquit
NRZ	Non-Return-To-Zero
OBC	On-Board Computer
PCB	Printed Circuit Board
PLL	Phase Locked Loop
PRBS	pseudorandom binary sequence
QoS	quality of service
RIU	Remote Interface Unit
Rj	Random jitter
Rx	Receiver
SAS	Serial Attached SCSI
SerDes	Serializer/Deserializer
SpFi	SpaceFibre
SSC	Spread-Spectrum Clocking
SSMM	Solid State Mass Memory
Tj	Total jitter
Tx	Transmitter
UI	Unit Interval

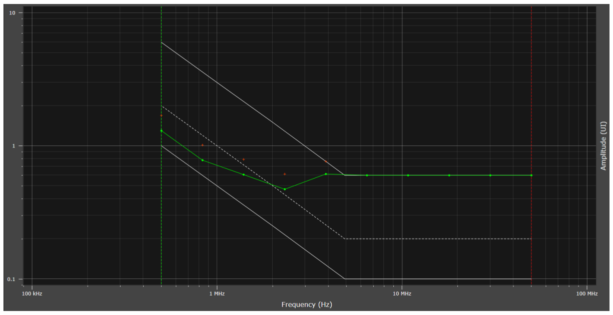


Fig. 12. Jitter tolerance measurements with USB 3.0 5G pre-set curves.

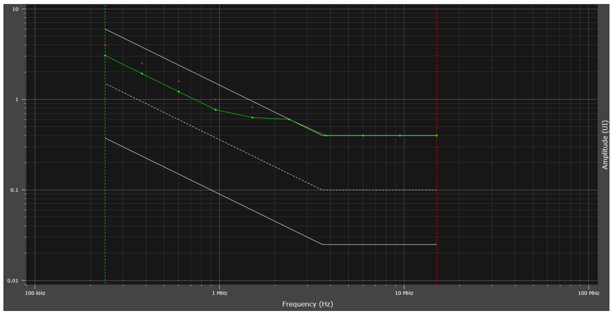


Fig. 13. Jitter tolerance measurements with SAS2 6G no SSC pre-set curves.

An Optical SpaceFibre Testbed for Transceiver Evaluation and Validation of a Design-Time Configurable Router

1st Malte Bargholz
On-board Computer and
Data Handling Section
European Space Agency, ESTEC
Noordwijk, The Netherlands
malte.bargholz@esa.int

2nd Wolfgang Rönninger
Microelectronics Section
European Space Agency, ESTEC
Noordwijk, The Netherlands
wolfgang.roenninger@esa.int

3rd Felix Siegle
On-board Computer and
Data Handling Section
European Space Agency, ESTEC
Noordwijk, The Netherlands
felix.siegle@esa.int

***Abstract*—State-of-the-art on-board data handling systems demand very high-speed serial communication links to avoid bottlenecks in data processing. These links can be implemented using optical transceivers in conjunction with high-speed Serializer/Deserializers (SERDES). On top of that, a high-speed protocol such as SpaceFibre provides the required system-level Fault-Detection, Fault-Isolation and Recovery Techniques (FDIR) and Quality of Service (QoS) guarantees. Optical transceivers provide several key advantages in comparison to classical electrical coupling. They allow for significant harness reduction while also providing data rates beyond 10 Gbit/s and superior characteristics such as galvanic isolation and high signal quality over long distances. Recently, companies such as Smith Interconnect or Glenair have successfully qualified optical transceivers for space applications, paving the way for usage in next-generation spacecraft. Current missions have also adopted multi-port and multi-lane link implementations, which utilize multiple such links in parallel to deal with increasing data throughput. This allows system designers to scale the available bandwidth to fulfil mission requirements. In this paper, we propose an FPGA-based optical transceiver testbed for SpaceFibre that comprises eight physical links at very high data rates of 10 Gbit/s to develop and validate both multi-port/multi-lane implementations and optical transceivers. We use this testbed to validate a SpaceFibre router. The router IP is generated by a customizable code generator, which derives the switch matrix from a system-level network topology description at design time. We show that this allows for a highly optimized implementation for modern space FPGAs and provides static fault isolation and security guarantees for the end-user.**

***Index Terms*—High-speed links, Optical transceivers, Testing, SpaceFibre, Routing, Design-time optimizations**

I. Introduction

In recent years, requirements for on-board data handling systems have increased significantly. When Envisat launched in 2002, state-of-the-art payload transmitters allowed downlink rates of up to $100\,\mathrm{Mbit/s}$ [5]. Twenty years later, optical laser terminals have increased the available data rate by at least one order of magnitude, with current implementations reaching multiple gigabits per second [9]. Additionally, such modules are available in increasingly smaller form factors and at low cost, with even Cubesat-sized terminals providing gigabits per second down link rate [4]. Similarly, current multi-spectral image sensors [6] require the handling of multiple channels of data with each channel reaching gigabits per second data rates. Current, and future, data handling systems must therefore be able to handle and process multiple of such gigabits per second data sinks and sources. To interface payloads and communication system at their respective data rates, the on-board data handling system requires very high-speed links capable of such speeds. These links can either be of parallel or serial nature, with the latter being favored due to the decreased implementation complexity with regards to signal skew and SWaP parameters. To facilitate the serialisation at the required data rates, implementations make use of high-speed SERDES that are available as dedicated ICs (e.g. TLK-2711) and as hard-macros in FPGA and System on Chip (SoC). On top of the SERDES, a protocol is employed that varies depending on the mission requirements and heritage. While WizardLink protocol implementations are common in European missions in recent years, US-based solutions are often based on Serial RapidIO. Both can provide multiple gigabits per second of data rate and Serial RapidIO additionally provides flow-control and FDIR by design, whereas WizardLink requires an additional high-level protocol to achieve this. SpaceFibre, the successor to the widely used SpaceWire, is another protocol option to provide data rates of multiple gigabits per second with built-in FDIR, QoS and deterministic communication functionality. In comparison to Serial RapidIO, SpaceFibre provides first-class support for mixed-criticality networks, which is an important step towards harness and complexity reduction [1].

At the required data rates of future on-board communication systems, signal integrity becomes an important design consideration. The serial nature of the links leads to a high switching frequency of the line, which limits the maximum length of copper harness before a significant signal integrity loss is observed. Optical fibre, on the other hand, provides high signal-to-noise ratios even over hundreds of meters of harness [8].

In addition, with increased on-board connectivity it is desirable to prevent fault propagation across link boundaries.

Copper links by nature electrically connect both terminals of a link and therefore easily propagate faults from one end of the link to the other. Optical links, on the other hand, provide galvanic isolation by design, and thus limit the failure propagation across links.

Finally, with harnesses of big satellites reaching hundred kilograms [1], harness reduction is a major focus of future satellite developments. Copper links are significantly heavier and have an increased form factor in terms of connectors than comparable optical links, which often combine multiple links into a single fibre connection.

In summary, optical-based high-speed links are superior to copper links with respect to harness size, weight and complexity, signal integrity and in terms of fault isolation. They are therefore a promising target for future on-board communication links.

The paper is structured as follows: Section II discusses current optical transceivers, and presents the optical link testbed, which was developed to evaluate them for use in future on-board data-handling systems. In Section III we then present a novel design-time configurable SpaceFibre router framework that we evaluate using this testbed. Section IV summarizes the results and outlines future extensions of the testbed.

II. Optical Link Testbed

To best evaluate the potential use of optical transceivers in future on-board data handling systems, we present an FPGA-based testbed, which combines the widely used SpaceFibre protocol with state-of-the-art optical transceivers that have a space-grade equivalent. In short, we propose a testbed containing two FPGAs, one acting as the data source and sink, and the other one as data loopback or data processing/routing function. For the FPGA, we make use of the Xilinx XCKU060 and XCKU040, which are part of the Xilinx Kintex Ultrascale platform that has been proposed as the next space-qualified FPGA for future on-board data-handling units. The FPGAs are connected to each other using optical transceivers, which are in turn connected via optical fibers and to the their internal SERDES macros for data serialization/de-serialization. Internally, these macros are controlled by multiple SpaceFibre Codecs, which implement the high-level communication protocol. In total, eight of such codecs are employed, in order to emulate a wide range of on-board data handling network topologies. Both FPGAs are controlled using an attached lab computer, which facilitates experiments through a script interface. In the following sections, we first discuss the selection of optical transceivers (Section II-A), then describe the architecture of the testbed in detail (Section II-B), and finally discuss how the testbed was evaluated (Section II-D).

A. Space-grade optical transceivers

The market for optical transceivers is broad, both aerospace-grade and industrial parts through COTS spin-in are potentially viable for future missions.

Glenair offers a multiple optical transceivers, both for use in space and aerospace. The DataStar (tm) SPACE Quad Parallel optical transceiver offers $10\,\mathrm{Gbit/s}$ per lane and up-to 4 lanes per module. They are available for extended temperature ranges and are certified according to MIL-STD for vibration and shock. Additionally, they were tested up to a dose of $250\,\mathrm{krad}$ with no errors recorded. Proton and heavy-ion irradiation results are available on request.

Smith Interconnect (former Reflex Photonics) also offers (aero)space-grade optical transceivers within their SpaceAble product line. They offer both $10\,\mathrm{Gbit/s}$ and $28\,\mathrm{Gbit/s}$ lane-speed, with up-to 4 lanes per module. The modules are vibration, shock and temperature shock tested according to MIL-STD, and are qualified to work over an extended temperature range. Additionally, TID, proton and heavy-ion irradiation results are available on request.

The UK-based APITech also offers (aero)space-grade optical transceivers within their OptoFire (tm) series. Up-to 4 lanes of $10\,\mathrm{Gbit/s}$ are offered by a single OptoFire module, which is able to operate in an extended temperature profile. The modules also include radiation tolerance circuitry for harsh environments. Additionally, the OptoFire modules are free of ITAR restrictions.

On the COTS-side, Samtec offers the FireFly (tm) optical transceivers, which exist in 10 and $28\,\mathrm{Gbit/s}$ per lane and up-to 12 lanes per module. These transceivers have an extended temperature profile suitable for space and are tested for vibration and shock resilience according to MIL-STD801G. However, almost no public radiation results exists, with only the non-extended temperature range optical engine being tested recently for usage in the next-generation CERN Muon detector [2].

ESA has procured the Smith Interconnect LightAble series 10G LM and as such they are used in the first iteration of the optical transceiver testbed. However, all of the options presented above are candidates for usage in future missions. The testbed was designed in a way that allows testing different optical transceivers without major adaptation work.

B. Architecture

Figure 1 shows the architecture of the optical transceiver testbed. It consists of two FPGAs, a primary FPGA which acts a the data source and sink of the testbed, and a secondary FPGA, which contains a possible Design under Test (DUT) to process the data.

Both are equipped with a Smith Interconnect VITA-57.1 FMC daughter card that holds the LightAble SL 10G LM Transmission (TX)/Receive (RX) modules. To maximize the available channels between the two FPGAs two LightAble modules are used per card that together provide a maximum 12 full-duplex links. The modules are connected optically with two OM3 ribbon fibers. On the FPGA-side they are connected through the FMC connector to each FPGAs SERDES blocks, which are in turn connected to the SpaceFibre Codec from ESA's IP Core Library. The FMC cards also contain a small clock generation integrated circuit (IC), which provides both the system clock and SERDES clock to the FPGA. To simplify the design process, no clock-domain crossings exist in the

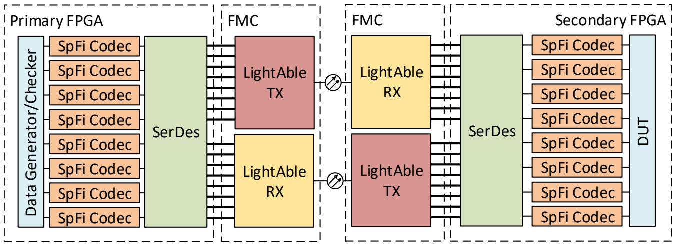


Fig. 1: A simplified block design of the optical transceiver testbed design.

design besides an elastic buffer between the receive-side of the SERDES and the SpaceFibre codec. All other design blocks run on the derived transmit clock of the SERDES.

On the primary side of the design the SpaceFibre Codecs are driven by a data generator and checker block, which sends configurable data sequences over the attached links and checks if they are received back correctly. The DUT is implemented on the secondary side and is similarly connected to the SpaceFibre Codecs. In the initial implementation only a basic DUT is implemented, which loops back the received data per link to validate the overall system design. The SpaceFibre Codec configuration and status register, the data-generator/checker and the DUT are connected to an APB Bus (not shown), which in turn is controlled by an UART AHB bus-master. The evaluation environment makes use of this interface to configure and monitor the relevant design blocks during test execution.

C. Implementation

For the implementation of the previously described design, two Xilinx Kintex Ultrascale series FPGA were used. A XCKU060 on a Hitech HTG-K800 development board implements the primary FPGA and a XCKU040 on a Xilinx KCU105 development boards implements the secondary FPGA. Due to limitations in the FMC wiring on the KCU105 board, only 8 links of the maximum 12 links are used in the current implementation. For each of the links the SpaceFibre codec is configured to provide 3 virtual channels, to make sure the channel arbitration logic is not optimized out. The Xilinx GTH SERDES blocks were configured for a line-rate of $10\,\mathrm{Gbit/s}$, which translates to an effective transmit clock of 250 MHz that is used as the primary system clock. A seperate 100 MHz clock provided by the FMC card is used for SERDES and design bring-up. For eight total links, two GTH quads are used per FPGA that share a PLL for deriving the transmit clock and to perform receive clock recovery. Both quads are further configured in single-lane mode, meaning that they share a transmit clock, but provide separate recovered receive clocks, which simplifies sharing the data-generation block across multiple links. The design is implemented in VHDL and is shared across both the primary and secondary FPGA, with a design-time generic selecting whether or not a DUT or data-generator is implemented.

TABLE I: Design utilization for XCKU060 target.

Block	LUTs	FF	BRAM
SpFi Codec	3627 (1.1 %)	2364 (0.4 %)	4 (0.4 %)
Data Generator	7324 (1.7 %)	7457 (0.9 %)	0 (0.0 %)
Top Level	37479 (11.3 %)	28160 (4.3 %)	36 (3.0 %)

D. Evaluation

The discussed design is implemented for specified line-rate of the optical transceivers ($10\,\mathrm{Gbit/s}$) in Vivado 2020.1. After successful implementation we recorded the resource utilization, critical paths and estimated power consumption. Since both targets share most of the VHDL code only the XCKU060 is evaluated below, but the results are, with slight deviation, also valid for the XCKU040 implementation.

Table I shows the design utilization of the design implemented for the XCKU060 target, which acts as the primary FPGA. Utilization of different blocks of the design are shown both in terms of absolute utilization and the percentage of the total FPGA resources. Most notably, the SpaceFibre Codec requires less than a percent of the FPGA resources in terms of lookup-table (LUT), Registers and block random-access memory (BRAM). In total, the eight SpaceFibre Codecs and data generator take about 10.5 % of the FPGAs LUTs, 4.1 % of Registers and 3 % of BRAM, the rest is taken up by glue-logic and the AHB/APB bus. This leaves ample space for the eventual experiments that will be implemented on the secondary FPGA.

The design was routed successfully on both targets, with the XCKU060 having the smaller setup and hold margin of 0.002 ns and 0.023 ns respectively. Inspection shows that the critical path lies in the SpaceFibre Codec, specifically in the CRC16 calculation that currently happens in the retry layer.

To estimate the power consumption of the design the Vivado Power Estimator is used, which is supplied a testbench generated stimulation activity file to improve estimation confidence. In the simulation all links are brought up and have the data generator enabled. Table II present the estimates power usage per design block. In total $2.069\,\mathrm{W}$ of power are consumed by the design, not including the power used by the optical transceiver or clock generation IC. Most of the power is consumed by the SERDES hard-macro, which comes in at 1.638 W or approximately 80 % of the total design power. This consumption is two order of magnitude higher than the power consumed by the SpaceFibre Codec implementation, that consumes about $0.047\,\mathrm{W}$. Taking the optical transceiver power of 100 mW per channel into account, the total consumption per link can be estimated to be approximately $P_{\text{link}} = 0.1W + \frac{1.638W}{8} + 0.047W \approx 0.35W$

TABLE II: Design Power Estimation for XCKU060 Target.

Block	SpFi Codec	Datagen	SERDES	Top Level
Power Usage [W]	0.047	0.056	1.638	2.069

As previously discussed, the design can be monitored and configured through the AHB bus, in conjunction with a PC application that communicates over USB-UART with the AHB master. To assess the link quality, a set of scripts for said application was developed that configured the data generator to continuously send data over all links and monitor if any errors occurred. No errors were recorded over a period of 24 hours with all links running at full speed, which correspond to 6912 Terabit of data sent error-free. Using the Poisson distribution, we can calculate the confidence level that the true bit-error rate (BER) of the system is below a specified BER of 1×10^{-15} using (1) (with $\text{BER}_s = 1 \times 10^{-15}$) to be 99.9 %.

$$CL = 1 - e^{-N*\text{BER}_s} * \sum_{k=0}^{E} \frac{(N * \text{BER}_s)^k}{k!} \tag{1}$$

III. Design-time configurable Router

With these newly introduced high-speed links on spacecraft, the amount of network traffic which needs to be handled poses a major challenge for the inter module connection network. Furthermore, with the ongoing miniaturization of on-board components it is feasible to add more and more high-performance instruments and processing onto future missions. This adds to the complexity of the overall network topology with multiple routing switches, which are connected together. As a concrete example of a simple *SpaceFibre* network we use the network in fig. 2. The `On-Board Computer (OBC)` orchestrates multiple `Instruments` which are transferring science-data into a `Mass-Memory-Unit (MMU)`. The components are connected with a *SpaceFibre* router.

The ECSS-E-ST-50-11C[10] lays out the overall architecture and operational principle of the *SpaceFibre* network layer, including the operational and high-level architecture requirements of a network router (ECSS-E-ST-50-11C: Figure 5-55). The requirements can be partitioned into four different parts for a concrete implementation:

- *Ports/Codecs*: Are translating the off-chip packet transmission into parallel on-chip signals.
- *Switch Matrix*: Is switching the packet in a on-chip network.
- *Routing Table*: Defines the routing between the ports.
- *Broadcast Mechanism*: Forwards and distributes broadcast messages.

One can imagine a general router implementation, utilizing a full crossbar as the switch matrix to connect all virtual channels of each port together. This router architecture is then configured at run-time to implement the desired switching characteristic such as the logical address table and the binding of the virtual channels to virtual networks. Such an implementation, whilst highly performant, uses a lot of logical and routing resources, as each port needs to have a direct connection to each other port. This approach also can lead to unforeseen errors by the configuration of the design. The approach of using a full switch-matrix also has security implications as traffic from different virtual networks use the same physical connections.

When we look at the example from fig. 2 it becomes apparent that this full connectivity is often not needed in reality. The traffic patterns are mostly distinct from each other and have a single destination. So full switching capabilities are not needed. When the *SpaceFibre* router is implemented on an FPGA we can optimize the needed hardware resources by tailoring the switch fabric to the specific network position of the router. This takes advantage of the fact that the overall network architecture is defined during design time and generally does not change over the mission duration.

We propose therefore to generate a customized FPGA implementation and corresponding switch fabric (as shown in Figure 3) during design time by analysing the static configuration of a specific network router. The next sections contain the details of the register-transfer level (RTL) design of the virtual network switching fabric (Section III-A1) and broadcast mechanism (Section III-A2). The evaluation and integration of the switch design into the testing framework introduced in

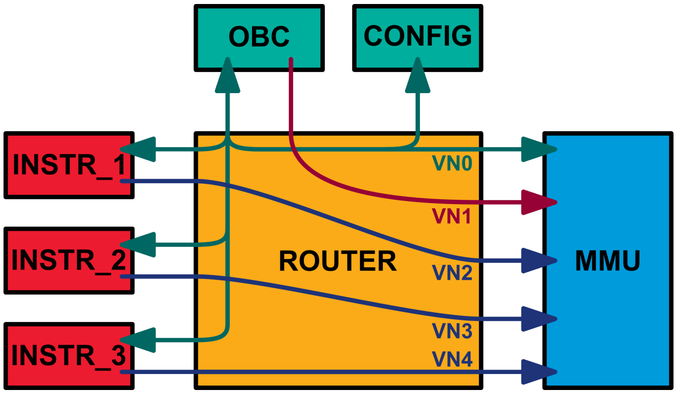


Fig. 2: Example of a *SpaceFibre* virtual network topology and traffic pattern. The `OBC` can configure the different components and issue *PUS* to the `MMU`. The instruments (`INSTR_1-3`) send their science data to the `MMU`. With *SpaceFibre* it is possible of having the network traffic virtually separated from each other. *VN0: Green*: Command & Control *VN1: Red*: *PUS* from `OBC` *VN2-4: Blue*: Science data from `INSTR_1-3`

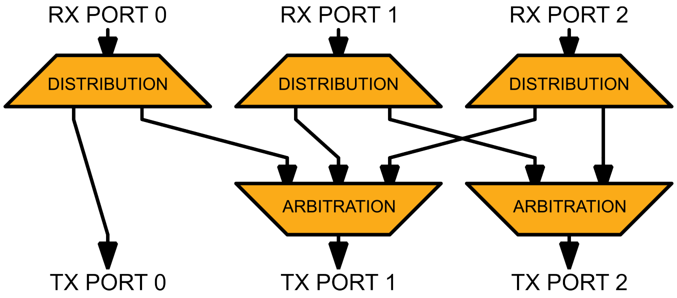


Fig. 3: Example of a custom fabric routing switch for a single virtual network. Physical connection can be optimized depending on the expected traffic inside the network.

section II-B is discussed in section III-C. We use the topology from fig. 2 as an example.

A. Architecture

The concrete routing switch fabric needed for *SpaceFibre* can be split into the two parts *Virtual Network Packet Switching* and *Broadcast Message Distribution*. The implementation is written in *SystemVerilog* and uses IPs of the open-source library *common_cells*[3] from the *PULP-Platform*[7] for the low-level data-plane stream management.

1) Routing Fabric Generator: The Hardware Description Language (HDL) for the customized routing switch fabric is generated with a generator written in PYTHON. The packet routing switch fabric topology is described in human-readable YAML format as shown in Listing 1. For the generator configuration the following information is needed:

- The name of the router
- The description of the ports
- A global logical address mapping (optional)
- Definition of the virtual networks and their connections

This parameters are mapped into the YAML format in the following way: The `name:` key describes the name of the router, which is used in the naming of the generated modules. The `ports:` key lists the used ports and their configuration parameters such as the number of virtual channels. The optional `addr_map_logic` key describes the global logical address mapping from address to port. In this example port index addressing is used. The `virtual_nets` key defines the virtual networks implemented in the switch matrix. Specifically, name and index of the virtual network, a list of participating ports and the existing connections between them are defined. For each connection the supported address decoding is also specified.

```yaml
name: spfi_router

ports:
  - name: port_name
    params:
      NumVc: number of virtual channels

addr_map_logic:
  - addr: port_name

virtual_nets:
  - name: vn_name
    glob_index: vn_index
    port_vc_map:
      port_name_1: vc_index
      port_name_2: vc_index
    connections:
      - from_port: port_name_1
        to_ports: [port_name_2]
        addr_type: ['path', 'logic']
```

Listing 1: YAML schema of the routing switch matrix generator.

From this information the generator builds an internal model of the routing switch and then generates the HDL code with templates. The generated code uses hand-written sub-instances and constructs. This increases the readability of the generated code and helps with the verification effort as the small instantiated sub-IPs can be easily verified separately to a high degree.

The generated structure for each individual virtual network is split into three parts; the *RX Packet Distribution* (Figure 4), *TX Packet Arbitration* (Figure 5) and the connections between them. The flow control happens on the fly, powered by the stream handshake semantics. This allows for dynamic back-pressure if a destination is not ready to process a stream item in the current clock-cycle. Therefore the fabric itself does not use large data buffers for holding complete packets. The switch fabric contains pipeline stages at the input and output to the RX and TX FIFOs as well as a pipeline stage in the connections between the distribution and arbitration. Therefore the latency of a packet is kept to a minimum.

a) RX Packet Distribution: The packet distribution pipeline is instantiated after the `RX FIFO` of a given `VC`. It takes care of the receive timeout functionality and address decoding to route the packet to the respective destination.

- `RX Timeout`: The RX timeout is responsible to prevent in-flight packets blocking the network when the originator of a packet stops sending the packet unexpectedly. A counter keeps track of the time since the last item of a packet was received. The counter is reset every time a packet item transaction happens. When the counter expires an EEP is sent further down the pipeline and all subsequent packet items are dropped until the RX transmits am EOP/EEP.
- `Address Extraction`: This module is responsible of extracting the address of the packet. The address is always extracted regardless of configuration. The module looks at the first item of a packet and extracts the address depending on the presence of fill characters. The extraction process takes one cycle as the address is latched into a flip-flop. This is to cut the combinational path through the address decoding. When an EOP/EEP is transferred through the module the address extraction gets primed for the next packet.
- `Address Decoding & Demultiplexing`: The extracted address is then used to determine the validity of the address. The global configuration determines the address map which is implemented as a constant lookup-table. This allows for the synthesis to optimize the translation to a routing index efficiently. The generator takes care of optimizing the address decoding rule list, depending on the ports which are actually connected as defined in the generator configuration file. When the address decoding is valid, a connection index is generated which steers the packet items towards the

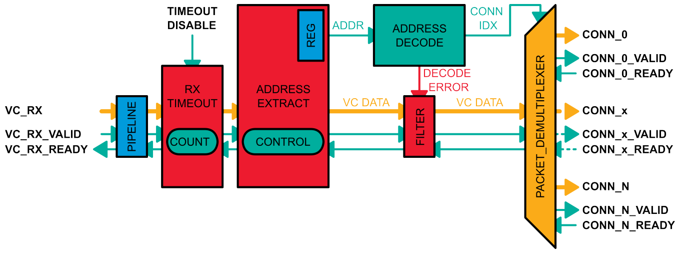


Fig. 4: Distribution pipeline instantiated after each virtual channel RX FIFO. It takes care of the receive timeout and packet demultiplexing towards the arbitration.

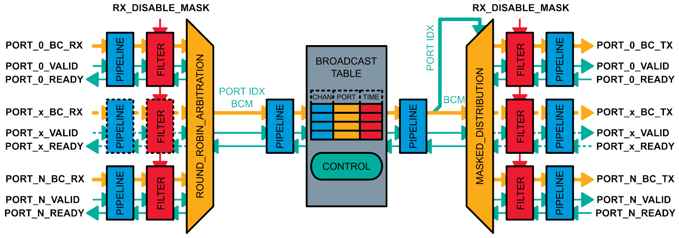


Fig. 6: Broadcast message pipeline architecture. Incoming broadcast messages are arbitrated, checked if legal to distribute and passed onto the transmit side.

defined TX structure. When an address is not mapped, a decode error is generated and the packet is dropped.

b) TX Packet Arbitration: The arbitration mechanism uses Round-Robin to merge packets from different RX sources. The arbitration tree looks at incoming packets when idle and selects a packet depending on its internal state. When a packet is finished the next one is selected.

- `Arbitration`: A packet from all incoming connections is selected using a round-robin scheme. For this a tree like structure is used to find out the next waiting packet. With the result a multiplexer on the data plane is configured until the last item of the packet stream is transferred.
- `TX Timeout`: The TX timeout mechanism makes sure that a packet can be actually transferred further down the TX chain. As with the RX timeout the TX contains a counter which keeps track of the clock cycles which have passed since the last packet item was successfully transferred. Then, if there is a valid packet item ready to be sent further but the TX side stalls, the counter will trigger a timeout. The module then tries to send an EEP and will start dropping all packets which are routed towards it. This is to prevent packets stalling out other TX destinations on the switching matrix due to a non-responsive TX port, as otherwise a stuck packet could prevent the flow of packets to other destinations. When the TX port is starting to accept packets again the next new packet will be transferred normally.

2) Broadcast Mechanism: The broadcast mechanism in Space-Fibre operates on individual messages and not on whole packet streams. This makes the design of the network considerably more straight forward. The implemented architecture block-diagram is given in fig. 6. However the possibility of so called *broadcast storms* needs to be accounted for and mitigated. For this a centralized broadcast table (Figure 7) which stores metadata about received broadcast messages is used.

The overall broadcast network topology is different and more regular than the one for the virtual channels. As the structure is regular, the instantiation of the components is done directly in HDL and does not use a generator, but generics to define the topology. This is because per port there exists exactly one broadcast interface and it is mandatory to provide broadcast functionality. With the help of stream filters individual RX or TX ports can be disabled.

The arbitration of the messages is again done using round-robin and provides the index of the port, at which the current broadcast message was received on. This port index is then used in the broadcast table as a tag. The Space-Fibre standard lays out how and when broadcast messages should be distributed to all other ports:

- The router comes out of reset, the message is sent further.
- If the port of arrival is the same for a previously seen broadcast channel, the message is sent further.

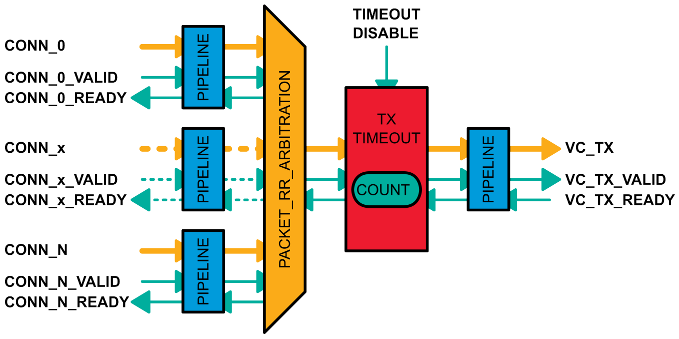


Fig. 5: Arbitration pipeline instantiated before each virtual channel TX FIFO. The packets are arbitrated using round-robin. The transmit timeout makes sure no packets can block the virtual network for other ports when one stops transmitting messages.

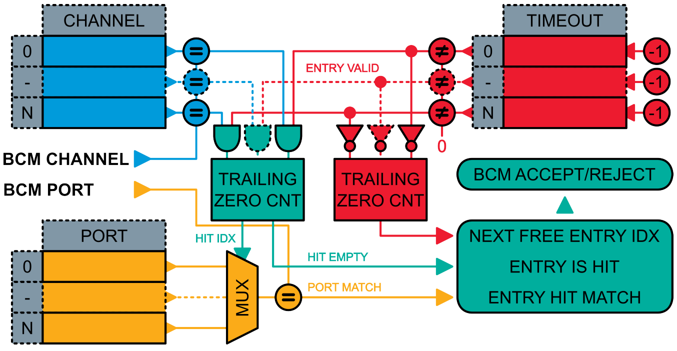


Fig. 7: The cache like broadcast table architecture for holding the metadata of recently seen broadcast messages.

TABLE III: Utilization of different sub-blocks of the router IP of the system from Figure 2 for a XCKU040 target (250 MHz).

(a) Optimized

Block	LUT	FF	BRAM
Switch	3622 (1.5 %)	4749 (1.0 %)	0 (0.0 %)
Broadcast	1187 (0.5 %)	2501 (0.5 %)	0 (0.0 %)
Codecs	19710 (8.1 %)	13932 (2.9 %)	34 (5.7 %)
Total	24802 (10.2 %)	22110 (4.6 %)	34 (5.7 %)

(b) Fully connected

Block	LUT	FF	BRAM
Switch	17740 (7.3 %)	24900 (5.1 %)	0 (0.0 %)
Broadcast	1158 (0.5 %)	2501 (0.5 %)	0 (0.0 %)
Codecs	27272 (11.3 %)	20487 (4.2 %)	66 (11.0 %)
Total	46492 (19.2 %)	48816 (10.1 %)	66 (11.0 %)

- If the port of arrival differs for a previously seen broadcast channel, a timeout needs to have been expired for the message to be accepted.

These requirements are to prevent the *broadcast storms* which can happen when there are circular paths in a network. This implementation uses a cache tag like structure, depicted in fig. 7, to keep track of the currently active in-flight broadcast channel messages.

Every time a broadcast message arrives at the table, a lookup is performed. When the channel is not present in the table the message is accepted and a entry added to the next free space in the table. For this the channel and incoming port index are saved. For the line also a counter is primed, which counts down continuously towards zero. As long as the counter has a non-zero value in it the table entry is valid. When the lookup finds a valid entry for the channel, the port of arrival is compared. The message is then only accepted if the arrival port and the saved one match. The matching of the channel and the pointer generation into the next free entry are done with trailing zero counters. Accepted messages are sent further to the distribution, rejected messages are dropped. The distribution will then send the message to all ports except of the port of arrival.

B. Validation

Both the packet routing and the broadcast mechanism are validated using constrained random verification powered by separate UVM-testbenches. For the switching fabric the testbench is generated together with the HDL. It generates for each RX input to the switch fabric a stream of random packets which are pushed into the fabric. The monitors generate a trace of all packets that enter and leave the switch fabric during simulation. A topology aware PYTHON script analyses these traces post-simulation and checks that the packets which entered a port are forwarded to the correct port or dropped depending on the configuration. This checking reuses the internal topology representation of the HDL generator.

The test-bench of the broadcast mechanism is hand-written, as the broadcast follows a more regular structure. Here the validation is performed in the test-bench during simulation using a scoreboard.

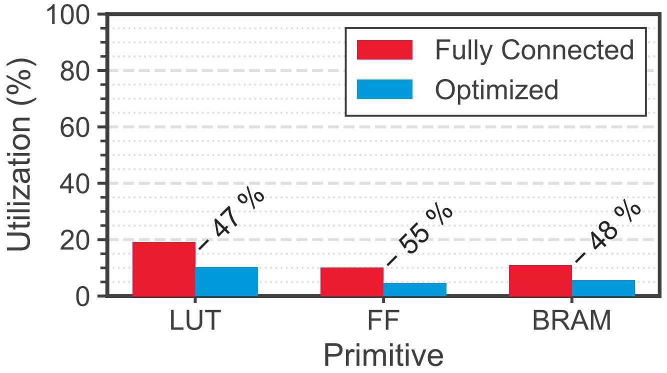


Fig. 8: Overall utilization of the router IP in different implementation variants for the XCKU040 target (250 MHz)

C. Evaluation

The generated *SpaceFibre* network and the broadcast mechanism are integrated into the optical transceiver testbed discussed in section II-B using the same design constraints as specified in section II-C. The generated router fabric is used instead of the simple loopback connections of the virtual channels and the broadcast distribution is connected to the broadcast ports of the codecs. The HDL generator also supports generation of the codec instantiation depending on the configuration. This generated file contains control and status registers for the routing fabric as well as an `APB` interconnect for accessing the registers of the codecs.

After integration the router functionality was briefly validated by enabling and configuring the data-generator block on the primary FPGA's side and recording if the expected traffic pattern was received.

Table III shows the utilization of the switch matrix, the broadcast mechanism, the SpaceFibre Codecs and the router top-level for different possible implementations of the network topology shown in Figure 2 split by FPGA primitives. Both absolute utilisation and relative utilisation to the total available primitives of the secondary FPGA are shown.

Specifically, Table IIIa demonstrates the utilisation of a fully optimized implementation that only generates the required virtual networks in the switch matrix, Table IIIb shows the utilisation of the same router but with a fully populated switch matrix. For the optimized version of the router the switch matrix only consumes about 1.5 % and 1 % of the secondary FPGA's LUT and FF resources respectively. The fully-connected router implementation, on the other hand, requires 7.3 % and 5.1 % of the FPGA's LUT and FF resources respectively, constituting an average increase of five times in terms of primitive usage. The broadcast part of the IP consumes a similar amount of resources for both configurations, which is in-line with expectations, given that in both variants all router ports expose a broadcast interface. It can be noted that no BRAM primitives are instantiated for the switch matrix

TABLE IV: Router power usage estimation per block in Watt for XCKU040 target (250 MHz).

	Block			
Variant	Switch Matrix	Broadcast	Codecs	Total
Fully Connected	0.18	0.02	0.78	0.98
Optimized	0.05	0.02	0.41	0.49

or the broadcast mechanism. Similar to the switch matrix, the instantiated SpaceFibre codecs consume more of the FPGA's resources in the fully populated variant of the implementation. Specifically, the SpaceFibre codecs consume an average of 1.5 times the primitives in the unoptimized, or fully-connected, variant. The biggest increase can be seen for BRAM usage, which almost doubles from 5.7 % to 11 %. We attribute this increase in primitive usage by the codecs to the fact that the HDL compiler is able to better optimize the code if only the required connections in the switch matrix are generated.

Figure 8 plots the overall utilization sorted by primitive of both implementation variants of the router top-level. The previously discussed decrease in primitive usage for both the switch matrix and the codec instantations in the optimized variant also translate into an overall decrease in utilization for the router top-level. On average the optimized variant of the router IP uses half of the resources taken up by a fully-connected router variant, which demonstrates the effectiveness of the design-time tailoring.

Finally, Table IV shows the estimated power consumption for both variants of the router implementation split by its fundamental design blocks. The estimation was derived using the Xilinx Vivado Power Estimator on the implemented design. Note that the Total column refers to the consumption of the router top-level, not the complete design. As with the utilization, the estimated power consumption varies only for the switch matrix and the codec instances. Overall, the power consumption of the router top-level can be roughly halved by applying the design-time tailoring. Taking into account the complete design including SERDES, the router only consumes about 18 % of the total power consumed for the optimized variant.

IV. Conclusion

Future on-board data-handling system will have to deal with increasing data rates both from sensors, memory units and downlinks, while also providing deterministic mixed-criticality routing with built-in fault isolation. We have shown that employing the SpaceFibre protocol over state-of-the-art optical transceivers is a viable option with the desired functional and non-functional properties. To aid its adoption in future missions we presented an FPGA-based optical testbed and used it to evaluate the Smith Interconnect LightAble LM10 series transceivers with SpaceFibre over multiple lanes. In total the FPGA implementation consumes less than 10 % of the FPGAs resources leaving ample space for potential DUTs, while showing a high data rate and small bit-error ratio.

Using the presented testbed, we then evaluated a design-time configurable SpaceFibre router. After validating its basic functionality using the testbed, we were able to show that customizing the switch matrix for a given network topology could reduce resource utilisation and power consumption by half, when compared to a full router implementation.

In the future we will extend the testbed capabilities with configurable per-lane data generators, which will allow us to automate and verify the routers security and isolation guarantees. Additionally, we intend to port the testbed to different optical transceivers to validate its flexibility. Finally, currently all lanes of the testbed operate independently from each other. A future extension could integrate a true multi-lane capable SpaceFibre codec to better mimic future on-board implementations.

References

[1] Dirk Thurnes. “Reduction of the Harness - The One Interface Illusion -reloaded-”. Presentation at 11th ESA workshop on Avionics, Data, Control and Software Systems (ADCSS2017). Oct. 2017.

[2] Rory O’Dwyer. “Radiation Hardness Assessment For Muon System Electronics Installed In The 2020 CMS Upgrade”. Bachelor’s thesis. Texas A&M University, May 2019.

[3] PULP-PLATFORM. *common_cells*. Version 1.24.1. Apr. 2022. URL: https://github.com/pulp-platform/common_cells (visited on 08/03/2022).

[4] *CUBECAT - Product Overview*. URL: https://www.fso-instruments.nl/cubecat/ (visited on 07/22/2021).

[5] *Envisat Overview*. URL: https://earth.esa.int/eogateway/missions/envisat/description (visited on 07/22/2021).

[6] *Orbis High Resolution Digital Output CMOS TDI Image Sensor*. URL: https://www.teledyneimaging.com/en/aerospace-and-defense/products/sensors-overview/cmos/orbis/ (visited on 07/22/2021).

[7] *PULP-PLATFORM*. ETH Zurich, University of Bologna. URL: https://pulp-platform.org/ (visited on 08/03/2022).

[8] *Smith Interconnect - LightABLE 10G LM Series 4TRX, 12TX, and 12RX, Product Brief*. URL: https://www.smithsinterconnect.com/products/optical-transceivers/embedded-transceivers/lightable-lm-40g-(full-duplex)-and-120g/RX (visited on 07/25/2021).

[9] *TESAT LCT135 Product Datasheet*. URL: https://www.tesat.de/images/tesat/products/220607_DataSheet_LCT135_A4.pdf (visited on 07/22/2021).

[10] *SpaceFibre – Very high-speed serial link*. Standard. Noordwijk, NL: European Cooperation for Space Standardization, May 2019. URL: https://ecss.nl/standard/ecss-e-st-50-11c-spacefibre-very-high-speed-serial-link/ (visited on 08/03/2022).

Definition, Implementation and Testing of an XML-based Packet Description Language for Traffic Analysis in a SpaceWire Communication

Roberto Ciardi
dept. of Information Engineering
University of Pisa
IngeniArs S.r.l.
Pisa, Italy
roberto.ciardi@phd.unipi.it
roberto.ciardi@ingeniars.com

Simone Vagaggini
dept. of Information Engineering
University of Pisa
IngeniArs S.r.l.
Pisa, Italy
simone.vagaggini@phd.unipi.it
simone.vagaggini@ingeniars.com

Antonino Marino
IngeniArs S.r.l.
Pisa, Italy
antonino.marino@ingeniars.com

Daniele Davalle
IngeniArs S.r.l.
Pisa, Italy
daniele.davalle@ingeniars.com

Gionata Benelli
IngeniArs S.r.l.
Pisa, Italy
gionata.benelli@ingeniars.com

Luca Fanucci
dept. of Information Engineering
University of Pisa
Pisa, Italy
luca.fanucci@unipi.it

***Abstract*—The on-board communication standard adopted in current generation space missions of the European Space Agency (ESA), and many other agencies as well, is SpaceWire (SpW). In a SpW network, data are exchanged as well-formed packets, which structure offers low packet overhead and allows developers to easily tailor their implementation for SpW applications. The flexible structure of SpW packets has allowed the definition of several SpW protocols, such as the Remote Memory Access Protocol (RMAP), that can be used in specific SpW applications. This paper presents the definition of the SpW Packet Description Language (SpW PDL), based on the eXtensible Markup Language (XML), to provide SpW developers with an instrument to easily define SpW packets, in both human-readable and machine-readable manner, aiding in the development of packet-format independent SpW applications. The SpW PDL allows the developer to simply define a single, or a set of, SpW packet structures by means of an XML file, adhering to a specifically designed XML schema. Such XML file can then be used by specific software for SpW-based systems.**

To verify the compliance with the SpW standard, it is critical to dispose of a set of tools and instrumentation to test SpW applications and SpW-based systems. Such equipment is part of the Electrical Ground Support Equipment (EGSE). In this scope, the SpW PDL described in this paper has been integrated into the SpaceART SpaceWire Sniffer, a user-friendly SpW link analyzer, designed to unobtrusively test and verify SpW communication between two SpW nodes. To prove the potential of the provided solution, a test-case in which SpW PDL is exploited for SpW traffic analysis is presented, demonstrating the benefits of the adoption of the SpW PDL in SpW systems test and verification.

The present work comprises five sections: at first, an introduction about EGSE, SpW Standard, and SpW packet format is given, introducing also the RMAP SpW protocol. Then the SpW PDL is presented, describing its definition and implementation. Later on, the SpaceART SpW Sniffer is described followed by a use-case scenario in which it is used to analyse SpW traffic, exploiting the SpW PDL. In conclusion, the result of the use of such technologies is given.

***Keywords*—SpaceWire (SpW), SpW packet, XML, Link Analyser, Sniffer, Elettrical Ground Support Equipment (EGSE)**

I. Introduction

On-Board-Data-Handling (OBDH) is a key feature of modern Spacecrafts, which usually includes various on-board devices that need to automatically communicate during the lifecycle of a space mission. The communication standard for OBDH, adopted in many space missions of the European Space Agency (ESA), and many other agencies as well, is SpaceWire (SpW) [1], [2], [3], [4]. SpW is based on point-to-point links in which data are exchanged in well-formatted SpW packets, as defined in the SpW standard [5]. The flexibility of the SpW packet structure allows to easily tailor the implementation of SpW communication for specific applications. On the other side, for each mission, unique packet formats are defined, and optimized for the specific mission requirements, making it hard to reuse defined SpW packet structures for future missions. A SpW packet format comprising a ”Protocol ID” field has been defined by the SpW working group as an extension of the existing packet format, to promote SpW packet identification and reuse [6], [7]. This has allowed the definition of SpW protocols such as the Remote Memory Access Protocol (RMAP), which can be used to effortlessly access memories on SpW nodes.

To test the functionality of specific SpW-based devices and verify the correctness of the communication, it is critical to dispose of specific instrumentation. Such instrumentation is part of the EGSE: a set of tools dedicated to testing and verification of the electrical functions of a spacecraft [8]. EGSE belongs to the Ground Segment of a space mission, thus the equipment is not intended to be launched on the spacecraft. It plays a critical role in both the design and test phases of a space mission, in particular during System Design and Assembly Integration and Test (AIT) operations. EGSE systems are also necessary at many different levels of testing and verification, from the single on-board device to the whole system.

An advanced EGSE system for SpW is the SpaceWire/SpaceFibre Analyser Real-Time (SpaceART) [9], [10], designed by IngeniArs S.r.l., that provides serial link communication interfaces for OBDH. The SpaceART SpaceWire Sniffer is a SpW Link Analyzer, based on SpaceART, which allows users to analyze SpW traffic between two nodes, identifying the exchanged SpW packets and their contents [11].

With this premise, IngeniArs S.r.l. has defined an XML-based SpW Packet Description Language (SpW PDL), described in this paper, to allow SpW users to define specific SpW packet structures in a human and machine-readable manner, to facilitate the SpW packet management. To ease the identification of specific packets in a SpW communication, which can include thousands of SpW packets, the SpW PDL has been integrated in the SpaceART SpW Sniffer. This paper addresses the implementation of the SpW PDL and its integration in the SpaceART SpW Sniffer, showing a use-case scenario of such technologies, to analyse the traffic of a SpW communication between two RMAP nodes.

In the next section, the SpaceWire standard is presented, focusing on the SpW packet structure and its Protocol ID extension. Later on, an overview of the SpaceART SpW Sniffer is given to introduce the reader to its use for testing SpW systems.

II. Related Work

A. SpaceWire Standard

SpaceWire (SpW) is the State-Of-The-Art concerning the spacecraft on-board communication link. SpW standard enables the communication among all the payload instrumentation, on-board computer, peripherals, and high data-rate sensors. SpW links are full-duplex bidirectional serial links, which can operate at data-rate from 2Mbps up to 400Mbps.

As explained in the standard definition [5], the SpW standard is a well-layered protocol, comprising: Physical Layer, Signal Layer, Character Layer, Exchange Layer, and Packet layer. A SpW packet has no limit on its size and is responsible for carrying the information between two SpW nodes. Fig. 1 shows the structure of a SpW packet, that comprises:

- The destination address: it represents either the identity of the destination node or the path that the packet has to take through a SpW network (to reach the destination node);
- The cargo: it is the data to be transferred;
- The End Of Packet (EOP): it signals the end of a packet.

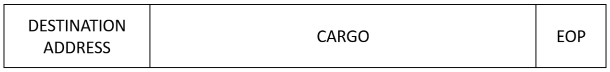
Other than SpW packets, some special characters can be exchanged over a SpW link as the Flow Control Token (FCT), which is sent every eight received data characters (i.e., part of a SpW packet), and the NULL character, which is sent to keep the link active, when there is no other information to send.

B. SpW Protocol ID and RMAP protocol

Although the standard SpW packet format is flexible and easy to use, it is not originally designed to promote protocol reuse. This forces SpW-based mission designers to define unique packet formats and how these packets are to be processed. For this reason, a new SpW packet format has been defined to extend the standard and support protocols development upon SpW [7], [6]. This has allowed to define a set of public SpW Protocols or define custom protocols for specific missions, to identify the packet contents and the associated processing, without replacing the standard format. This new format takes into consideration the first byte of the packet cargo, referred to as Protocol ID (Fig. 2), to recognize the used protocol. A subset of the defined public SpW protocols comprises the RMAP [12], the CCSDS Packet Transfer Protocol [13], the Reliable Data Delivery Protocol (RDDP).

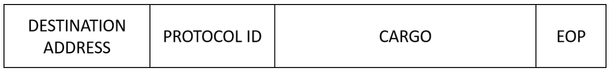
In particular, the RMAP is a SpW protocol designed to directly read from/write to memory inside a SpW node, across the SpaceWire network. The RMAP protocol aims to standardize how SpW units are configured and to provide a low-level mechanism for the transfer of data between two SpW nodes. In an RMAP communication a SpW node, called RMAP initiator, sends commands to read/write a memory present on a second SpW node, called RMAP target. Each SpW RMAP packet is composed of a header, comprising the RMAP protocol ID (01), and an eventual data payload. A simple Cyclic Redundancy Check (CRC) byte is computed for both header and payload and is appended respectively to the header and the payload. The RMAP packet sent by an initiator can be one of three types, Read, Write, or Read-Modify-Write, with the target that can send a reply that carries the command status and eventually the read data. Fig. 3 shows the format of a Read command which does not have a payload, hence its content is composed only by the header. The header shown in Fig. 3 is the same for all the RMAP command (whereas it slightly differs for the RMAP replies) and contains:

- the destination and source address (i.e., the logical or path addresses);
- the protocol ID (i.e., 01 for RMAP);
- a configuration byte to indicate some packet configuration as the type of the command (i.e., read);



DESTINATION ADDRESS	CARGO	EOP

Fig. 1. Standard SpW packet format.



DESTINATION ADDRESS	PROTOCOL ID	CARGO	EOP

Fig. 2. SpW packet format with Protocol ID field.

- a destination key field for authentication purpose;
- a transaction id that identifies the RMAP transaction, hence indicating the correspondence between RMAP command and RMAP reply;
- an extended address field used to extend the memory location address from/to which read/write;
- the memory location address from/to which read/write, on 4 bytes;
- the length of the data to be read/written, on 3 bytes;
- the header crc, computed as indicated in the standard [12].

When the Read command is sent, to read a specific memory location, the RMAP target validate its content (i.e., checks if the header CRC is valid or the memory location address and data length are valid, etc.). If the command is valid, the data are read from the memory and sent to the initiator in a Read Reply (Fig. 4). Such reply also has a header field, similar to the header of the RMAP requests, but carrying the status of the command instead of the destination key, and is followed by the data payload, which contains the bytes read from the memory. The CRC is computed for both header and data.

Similarly to Read commands, a Write command (which structure is not reported here) indicates in which memory location and how many data bytes must be written in the target memory. The payload of a Write command contains the data to be written in memory, followed by the data CRC. Differently from Read commands, the target does not reply to a Write command unless specified in the Write command itself, in the configuration byte of the header.

Finally, a Read-Modify-Write command is sent by the initiator to read a memory location and consequently write it in a single atomic operation.

This protocol as it is defined, is very useful for reading/writing memory on a remote SpW node, using a simple well-structured SpW packet. Applications exploiting the RMAP protocol can be developed to configure a SpaceWire network, control SpaceWire nodes, and to transfer data to and from SpaceWire nodes.

DESTINATION ADDRESS	PROTOCOL ID	CONFIG. BYTE	DESTINATION KEY
SOURCE ADDRESS	TRANSACTION ID (MSB)	TRANSACTION ID (LSB)	EXTENDED ADDR.
READ ADDRESS (MSB)	READ ADDRESS	READ ADDRESS	READ ADDRESS (LSB)
DATA LENGTH (MSB)	DATA LENGTH	DATA LENGTH	HEADER CRC
EOP			

Fig. 3. RMAP read command SpW packet format.

SOURCE ADDRESS	PROTOCOL ID	CONFIG. BYTE	STATUS
DESTINATION ADDRESS	TRANSACTION ID (MSB)	TRANSACTION ID (LSB)	reserved
DATA LENGTH (MSB)	DATA LENGTH	DATA LENGTH	HEADER CRC
Data (FIRST)	…	Data (LAST)	Data CRC
EOP			

Fig. 4. RMAP read reply SpW packet format.

For the purpose of this paper, an RMAP communication will be involved to test the SpW PDL described in this paper. The precise format of the RMAP packet will provide a perfect object to test the potential of the SpW PDL for reproducing the structure of such packet in a user-friendly manner.

C. SpaceART SpW Sniffer

The SpaceART SpW Sniffer [11], also referred to as Sniffer, is a link analyser part of the SpaceART products family [9] (Fig. 5). Specifically designed to support test and debug phases of SpW-based systems, the Sniffer allows the analysis of SpW traffic between two nodes at SpW character level, without interacting with the communication.

It is designed for those applications which require analysis of long SpW communications (hours) and advanced trigger conditions, to facilitate the acquisition of portions of interest. In particular, the Sniffer comprises 2 SpW ports – with a traffic speed up to 200 Mbps - compliant with the SpW standard, and an RX trace memory for each SpW interface. The presence of four SMA connectors allows using of external synchronization mechanisms that can be useful, for example, to synchronize time between different SpW units or to inject link disconnections. The data traffic flowing on each side of the SpW connection is stored by the Sniffer and sent to a Host-PC, connected through a 1 Gigabit-Ethernet interface. On the Host-PC, an easy-to-use Graphical User Interface (GUI) provides features to efficiently analyze the stored data. The SpW characters saved by the Sniffer are stored in a database file (i.e., can be stored for reuse) and can be visualized and navigated both at the packet level and character level.

The Sniffer represents a powerful instrument for SpW users who need to verify the correct functionalities of SpW-based equipment. The integration of the SpW PDL into the Sniffer, described below, for identifying specific SpW packets in the SpW traffic, enhances a user-friendly way of managing SpW

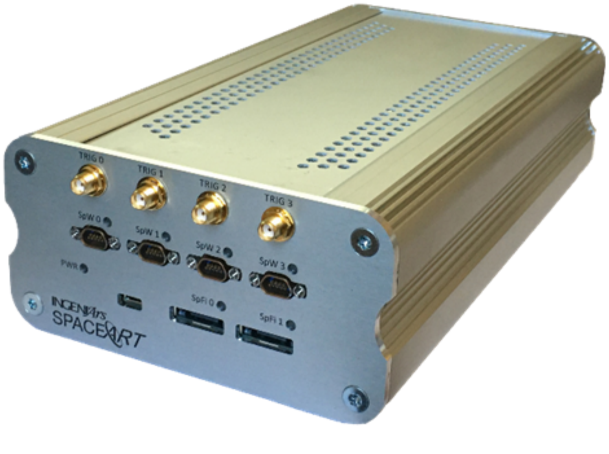


Fig. 5. SpaceART SpW Sniffer hardware unit.

packets and provides a simple and efficient tool for SpW traffic investigation.

III. SpW Packet Description Language

When dealing with SpW data traffic, SpW users usually need a mechanism to generate or parse well-structured SpW packets. Each space mission uses specifically formatted packets that are precisely tailored for the mission. For this reason, SpW developers usually need to develop a built-in special-purpose SpW packet manager, that handles Spw packet construction and decomposition, addressing every single type of packet needed for that specific mission. This process can be tedious, above all for such applications that require the use of several different packets or very long packets. In particular, the fixed structure of specific SpW packets is easy to handle for machines but not for humans who need to pay particular attention to packet dimension and each packet field, while developing the SpW communication for an application. Following the idea of defining an XML-based packet description language for network packet processing, given in [14], an XML-based SpW Packet Description Language was defined.

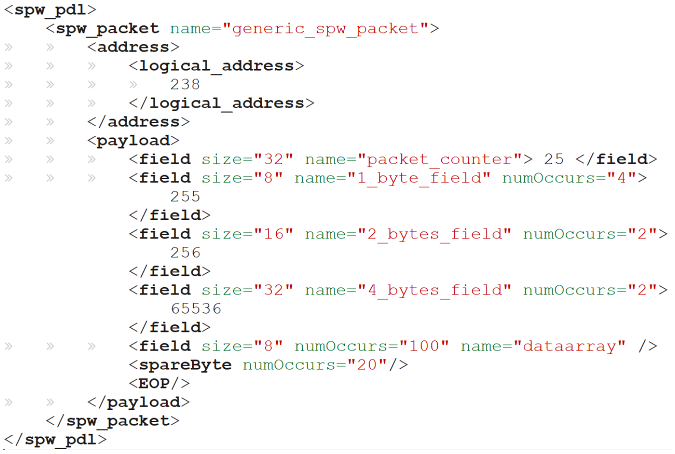
XML is a human and machine-readable language widely adopted especially in web applications because of its simplicity and flexibility. The SpW PDL allows describing one or more SpW packets in an XML file, to be processed by SpW applications (i.e., to generate SpW data or process and match received SpW character). The description of a SpW packet is based on the standard structure of a SpW packet (Fig. 1)) with an address, describing the address of the packet (e.g., logical address or path address), followed by a payload carrying the data. An example is provided in Fig. 6 where the description of a packet named ”generic spw packet” is given.

Following, the main tag fields of the SpW PDL, and their usage, are listed:

- *<spw_pdl>*: it is the root tag, indicating that the file contains the description of one or more SpW packets;
- *<spw_packet>*: this tag contains the structure of a single SpW packet and has two main children tags, the address tag and the payload tag. More than one <spw_packet> tags can appear in a SpW PDL file;
- *<address>*: this tag is a child of the <spw_packet> tag and contains the address of the SpW packet. The address is a child of this tag and can be a logical address, hence an integer value between 32 and 255, or a path address, hence a set of ports to be traversed by the packet;
- *<logical_address>*: this tag is a child of the <address> tag used alternatively to the <path_address> tag. It can contain an integer value between 32 and 255, indicating the logical destination address of the packet;
- *<path_address>*: this tag is a child of the <address> tag, used alternatively to the <logical_address>. It contains a set of children <port> tags to indicate the path address of the SpW packet;
- *<port>*: this tag is repeated as a child of the <path_address> to indicate the path, as a set of network ports, to be traversed by the SpW packet. This tag must contain an integer between 1 and 31;
- *<payload>*: this tag is a child of the <spw_packet> and contains the data of the packet as a set of <field> tags. The payload of the packet should be processed iteratively, with the field that are sorted as specified in the SpW PDL file;
- *<field>*: this tag is a child of the <payload> tag and represents a field of the packet. It has a mandatory ”size” attribute, that indicates the size in bits of the field. The size of each field can be used by a SpW PDL processor to compute the overall size of the packet. The user can use the <field> tag to define the structure of the packet, indicating each field composing the SpW packet. If a field is repeated an optional attribute ”numOccurs” can be used. The field can contain an integer value, indicating the value of that field in that SpW packet, or can be empty, depending on the application in which it should be used;
- *<spareByte>*: this tag is an optional child of the <payload> tag and can be used to indicate the presence of one or multiple (numOccurs > 1) spare byte, hence bytes whose value is not relevant;
- *<EOP>*: this tag is an optional child of the <payload> tag, indicating the end of the packet. If the tag is missing the payload ends with the last <field> or <spareByte> object;
- *<protocol_id>*: this is another optional tag, child of the <spw_packet>, that can be used for applications using a SpW protocol, to indicate the protocol ID of the SpW packet.

This XML-based language can be used in different manners to define the structure of one or multiple SpW packets.

The contents of the fields can be tailored for each specific application building complex or simple SpW PDLs based on the user's needs. For example, in an application for packet generation, a user should indicate the value contained in each field object of a SpW packet, that a properly-developed application would process to build the SpW packet content to



```
<spw_pdl>
    <spw_packet name="generic_spw_packet">
»   »   <address>
»   »   »   <logical_address>
»   »   »   »   238
»   »   »   </logical_address>
»   »   </address>
»   »   <payload>
»   »   »   <field size="32" name="packet_counter"> 25 </field>
»   »   »   <field size="8" name="1_byte_field" numOccurs="4">
                255
            </field>
            <field size="16" name="2_bytes_field" numOccurs="2">
                256
            </field>
            <field size="32" name="4_bytes_field" numOccurs="2">
                65536
            </field>
»   »   »   <field size="8" numOccurs="100" name="dataarray" />
            <spareByte numOccurs="20"/>
            <EOP/>
»   »   </payload>
    </spw_packet>
</spw_pdl>
```

Fig. 6. SpW PDL of a generic SpW packet.

```
<spw_pdl>
    <spw_packet name="RMAP_read_command">
»   »   <address>
»   »   »   <logical_address> 200 </logical_address>
»   »   </address>
        <protocol_id> 01 </protocol_id>
»   »   <payload>
            <field size="8" name="config_byte"> 76 </field>
            <field size="8" name="destination_key"> 00 </field>
            <field size="8" name="src_logical_addr"> 201 </field>
            <field size="16" name="transaction_id"> 01 </field>
            <field size="8" name="extended_addr"> 00 </field>
            <field size="32" name="read_address"> 00 </field>
            <field size="24" name="data_length"> 08 </field>
            <field size="8" name="header_crc"> 205 </field>
            <EOP/>
»   »   </payload>
    </spw_packet>
</spw_pdl>
```

Fig. 7. SpW PDL of an RMAP read command.

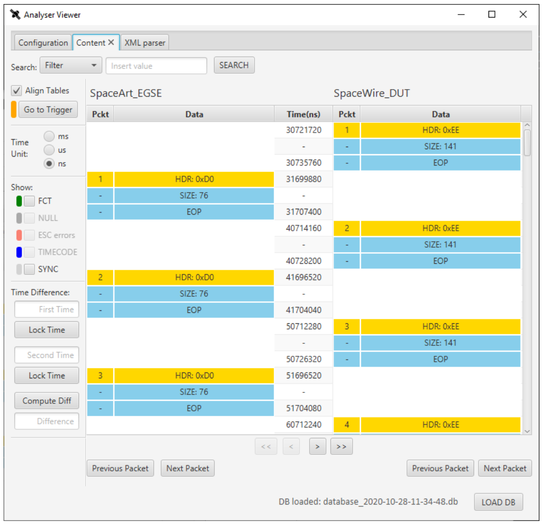
be sent over a SpW connection. Fig. 7 shows an example of a SpW PDL of a SpW packet of an RMAP Read command, with the structure shown in Fig. 3.The RMAP header is contained in the <payload> tag, with each field that is named as RMAP definition, and is given a value to perform the read request. In particular, such a packet would require to read 8 bytes from memory location 0. In an RMAP application, this XML file could be used to easily build multiple RMAP packets to be sent to an RMAP target.

An application using the SpW PDL is the SpaceART SpW Sniffer [11], in which the XML description of a packet can be used to identify the occurrence of that packet in the SpW data traffic. In this case, the SpW PDL description of a SpW packet can match the value of one or multiple fields of the packet, or can contain empty fields for matching all the packets that have the given structure. The next section shows the integration of the SpW PDL into the SpaceART SpW Sniffer, highlighting the capabilities of the proposed solution.

IV. SpW PDL Integration into SpaceART SpW Sniffer

The SpaceART SpW Sniffer is a Link Analyser to store and examine data traffic in a SpW communication. The Sniffer stores the ”sniffed” data into a database that can be explored, thanks to a GUI, to check the SpW data traffic and discover an eventual malfunction in the communication. Given that the Sniffer can store a large quantity of data, up to several minutes of SpW communication, it can be hard to find a specific set of characters or packets in the stored data. Although the GUI allows navigation of the data in a meaningful way, showing the SpW packets flowing through the connection and their contents, it may be hard to identify specific SpW packets. The definition of the SpW PDL allows using the XML description of a SpW packet, to individuate its occurrences in the SpW data flow. This will allow the user to investigate the data traffic in a user-friendly manner, verifying the SpW communication with the deserved accuracy.

In a generic SpW traffic acquisition, the user visualizes the data transmitted between two SpW nodes as a set of SpW packets with eventual SpW special characters (e.g., FCTs, NULLs, etc.). Fig. 8 shows a portion of the packet visualization of the Sniffer for a SpW traffic acquisition. The data traffic is visualized in two columns aligned on time. On the left column, it is possible to see the packets that are received by an end of the SpW link, whereas on the right we see the packets on the other end. This visualization indicates only the header and the size of each packet and is organized into several pages. In this case, the acquired database contains a large number of packets and the user can navigate through them. Double-clicking on a packet, the user can visualize the content of the packet (Fig. 9), hence the bytes that are part of the packet, from its header to the EOP. In this window, each SpW character is shown with its value as a decimal, hexadecimal, or 8 bits value. Although packet visualization is thought to be user-friendly, it is easy to understand that inspecting more packets can be a tedious operation for a user. For this reason, the use of the SpW PDL has been integrated with the GUI in an appropriate XML parser tab



Search: Filter ▾ Insert value

☑ Align Tables

Time Unit: ○ ms ○ us ● ns

Show: □ FCT □ NULL □ ESC errors □ TIMECODE □ SYNC

Time Difference: First Time, Second Time, Difference

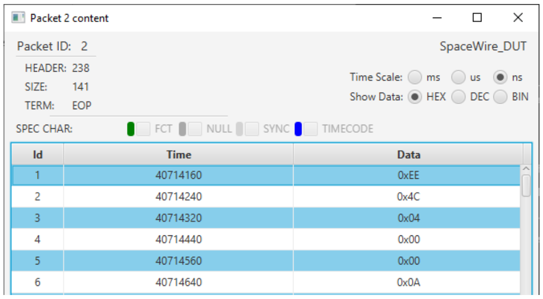
SpaceArt_EGSE — SpaceWire_DUT

Pckt	Data	Time(ns)	Pckt	Data
		30721720	1	HDR: 0xEE
		-	-	SIZE: 141
		30735760	-	EOP
1	HDR: 0xD0	31699880		
-	SIZE: 76	-		
-	EOP	31707400		
		40714160	2	HDR: 0xEE
		-	-	SIZE: 141
		40728200	-	EOP
2	HDR: 0xD0	41696520		
-	SIZE: 76	-		
-	EOP	41704040		
		50712280	3	HDR: 0xEE
		-	-	SIZE: 141
		50726320	-	EOP
3	HDR: 0xD0	51696520		
-	SIZE: 76	-		
-	EOP	51704080		
		60712240	4	HDR: 0xEE

DB loaded: database_2020-10-28-11-34-48.db

Fig. 8. Sniffer GUI visualization of data in a generic traffic acquisition.

The XML parser allows inserting a well-formatted XML file and then checking the stored SpW traffic to find the occurrences of the SpW packets defined in the file. Given the flexibility of the SpW packet format, the XML parser find the occurrence of a specific packet performing a match over the destination address of a packet, indicated in the *<address>* field, and over the size of the packet, either indicated with the



Packet ID: 2 — SpaceWire_DUT

HEADER: 238
SIZE: 141
TERM: EOP

Time Scale: ○ ms ○ us ● ns
Show Data: ● HEX ○ DEC ○ BIN

SPEC CHAR: □ FCT □ NULL □ SYNC □ TIMECODE

Id	Time	Data
1	40714160	0xEE
2	40714240	0x4C
3	40714320	0x04
4	40714440	0x00
5	40714560	0x00
6	40714640	0x0A

Fig. 9. Sniffer GUI visualization of the content of a generic SpW packet.

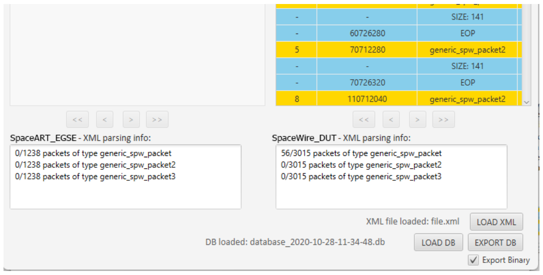
attribute ”size” of the <*payload*> field or computed as the sum of the sizes of each field of the payload (e.g., <*field*> or <*spareByte*>). If the packet size is not indicated in the SpW packet description, the XML parser matches only the destination address of the packet. If the value of one or more fields of the payload of the packet is given, the XML parser will check also that value. For example, given the SpW PDL in Fig. 6, the XML parser would match all those packets that have a logical address equal to 238 (0xEE), a size that is the payload size computed as the sum of the sizes of the fields (140 bytes) and each field values as given in the XML (i.e., 25 for the field ”packet_counter” with size 32). Also, the optional <*protocol_id*> field could be used to describe the packet, in such an application that makes use of the protocol id.

To give an example of the XML parser work, the SpW PDL given in Fig. 6 has been enriched with two other SpW packets description (not shown here), and the file has been used to inspect the data traffic visualized in Fig. 8. The result is given in Fig. 10 that shows that checking all the stored packets (1238 on channel A and 3015 on channel B), 56 packets of the type ”generic_spw_packet” have been identified, whereas the other types of packets are not present. Moreover, the packets can be visualized in the table and each packet can be opened to check its content, visualizing the data as structured in the XML file. In Fig. 9, the content of the packet was shown as the set of bytes composing the packet, in the case of the XML parser (Fig. 11) the content of the packet is shown as it is formatted in the XML file (Fig. 6). In this way, the user has direct access to the structured content of the packet and can check the value of structured fields in a simplified manner.

The next section describes the use of the Sniffer to test an RMAP communication between an RMAP initiator and an RMAP target. The SpW PDL will be utilized to describe well-structured RMAP packets and visualize their contents in the SpW data flow.

V. Test and Results

This section describes of a use-case scenario in which the SpaceART Sniffer is used to analyse an RMAP communication (Fig. 12). The test will show the benefits of using the SpW PDL in the case of a SpW communication based on well-structured SpW packets, such as the communication between an RMAP Initiator and an RMAP Target.



-	-	SIZE: 141
-	60726280	EOP
5	70712280	generic_spw_packet2
-	-	SIZE: 141
-	70726320	EOP
8	110712040	generic_spw_packet2

SpaceART_EGSE - XML parsing info:

```
0/1238 packets of type generic_spw_packet
0/1238 packets of type generic_spw_packet2
0/1238 packets of type generic_spw_packet3
```

SpaceWire_DUT - XML parsing info:

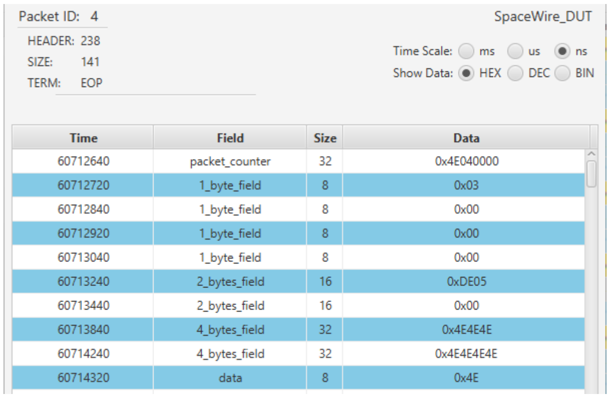
```
56/3015 packets of type generic_spw_packet
0/3015 packets of type generic_spw_packet2
0/3015 packets of type generic_spw_packet3
```

XML file loaded: file.xml

DB loaded: database_2020-10-28-11-34-48.db

☑ Export Binary

Fig. 10. Portion of Sniffer GUI visualization of XML tab where 56 packets have been identified by their XML structure.



Packet ID: 4 SpaceWire_DUT

HEADER: 238
SIZE: 141
TERM: EOP

Time Scale: ○ ms ○ us ● ns
Show Data: ● HEX ○ DEC ○ BIN

Time	Field	Size	Data
60712640	packet_counter	32	0x4E040000
60712720	1_byte_field	8	0x03
60712840	1_byte_field	8	0x00
60712920	1_byte_field	8	0x00
60713040	1_byte_field	8	0x00
60713240	2_bytes_field	16	0xDE05
60713440	2_bytes_field	16	0x00
60713840	4_bytes_field	32	0x4E4E4E
60714240	4_bytes_field	32	0x4E4E4E4E
60714320	data	8	0x4E

Fig. 11. Sniffer GUI visualization of the content of a packet structured as in the SpW PDL.

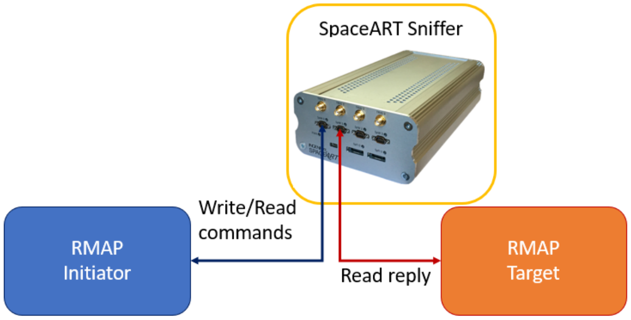
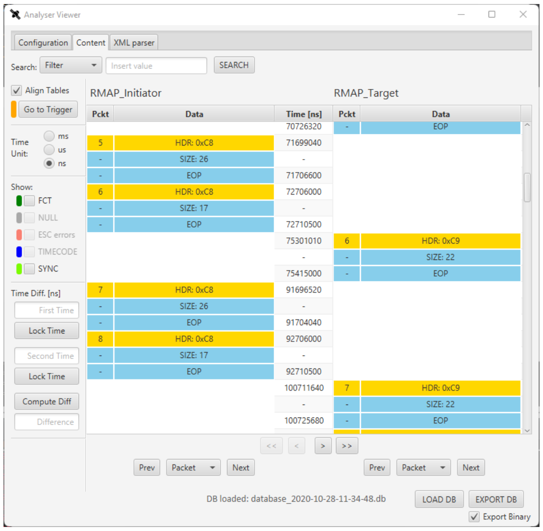


Fig. 12. RMAP communication test-scenario.

In this case, the RMAP Initiator sends a set of Write requests, each followed by a Read request, to Write some location and check that the location is correctly written. The RMAP Target does not reply to the Write requests but sends a reply to each Read request. The communication is synchronized so that the RMAP Initiator sends a new Write command only when the reply of a Read command is received. This behaviour is not mandatory for any RMAP communication, but it is useful in our case to show the data flow on the Sniffer. Eventually the RMAP Initiator and RMAP Target send some dummy packet, not related to the RMAP communication, for testing purposes. Fig. 13 shows a portion of the communication, as visualized in the Sniffer GUI, in which we can see the correspondence between the two packets sent from the RMAP Initiator (i.e., the Write and Read commands) and the packet sent by the RMAP Target (i.e., the Read reply) after the reception of the Initiator requests. The Read Read commands are composed only by the header (Fig. 3), whereas the Write commands and the Read replies (Fig. 4) contain also the data, hence the Write command is composed of 26 bytes (16 bytes of the header plus 8 bytes of data plus 1 byte for data CRC and the EOP),the Read command instead is composed of 17 bytes (16 bytes of the header plus the EOP). On the other side the RMAP reply packet (Fig. 4) is on 22 bytes (12 bytes of the header plus 8



Search: Filter | Insert value

☑ Align Tables

Time Unit: ○ ms ○ us ● ns

Show: □ FCT □ NULL □ ESC errors □ TIMECODE □ SYNC

Time Diff. [ns]: First Time | Second Time | Difference

RMAP_Initiator			RMAP_Target	
Pckt	Data	Time [ns]	Pckt	Data
		70726320	-	EOP
5	HDR: 0xC8	71699040		
-	SIZE: 26	-		
-	EOP	71706600		
6	HDR: 0xC8	72706000		
-	SIZE: 17	-		
-	EOP	72710500		
		75301010	6	HDR: 0xC9
		-	-	SIZE: 22
		75415000	-	EOP
7	HDR: 0xC8	91696520		
-	SIZE: 26	-		
-	EOP	91704040		
8	HDR: 0xC8	92706000		
-	SIZE: 17	-		
-	EOP	92710500		
		100711640	7	HDR: 0xC9
		-	-	SIZE: 22
		100725680	-	EOP

DB loaded: database_2020-10-28-11-34-48.db

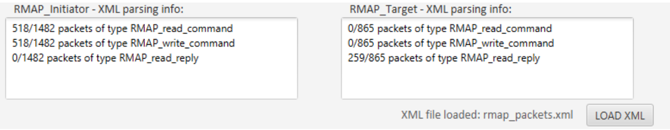
☑ Export Binary

Fig. 13. Sniffer GUI visualization of RMAP communication in the SpW data flow.

bytes of data plus 1 byte of data CRC plus the EOP).

In this scenario, as mentioned before, the RMAP Initiator and RMAP Target exchange several SpW packets, not only RMAP packets. In this configuration, we can understand how it can become hard to identify the RMAP packets in the data traffic: in presence of several packets (RMAP and non-RMAP) the user would need to open each packet and check the second byte (i.e., the Protocol ID), to check if the packet is an RMAP packet, and the bytes composing the RMAP header, to identify the type of the packet (i.e., Read, Write, Read-Modify-Write). Considering the large amount of packets that are exchanged, this would become a tedious task.

Defining an XML file as the one shown in Fig. 7, in which analogous packet descriptions can be added for Write and Read-Modify-Write commands and Read/Write replies, the user can use the XML parser to identify the occurrences of the RMAP packets. In particular, in the presented scenario, the XML parser identifies 518 Read and Write commands from the RMAP Initiator and 518 Read replies from the RMAP target, as depicted in Fig. 14, with the number of replies that matches the number of requests, as expected. We can notice that the number of found packets is a portion of the number of packets exchanged over the SpW link, but thanks to the XML parser it is easy to identify the packet occurrences. Moreover, by expanding SpW packet content, the user can visualize the content structured as the RMAP packet, simplifying the process of analysis of the RMAP communication. Fig. 15 shows the content of an RMAP Read reply packet sent by the RMAP target. Thanks to the use of the SpW PDL the structure of the content is the one shown in Fig. 4: we can notice the protocol id field as the first field of the packet, followed by the other fields of the RMAP header, for example, the 3-bytes data length (0x08). The header is then followed by the data that has been required by the RMAP initiator and that the RMAP target reads from the memory of the SpW node. This visualization, in contrast with the one shown in Fig. 9, shows how the XML structure can help the user in the data analysis, providing the SpW packet not as a simple set of bytes but as well-formatted structured information.



RMAP_Initiator - XML parsing info:

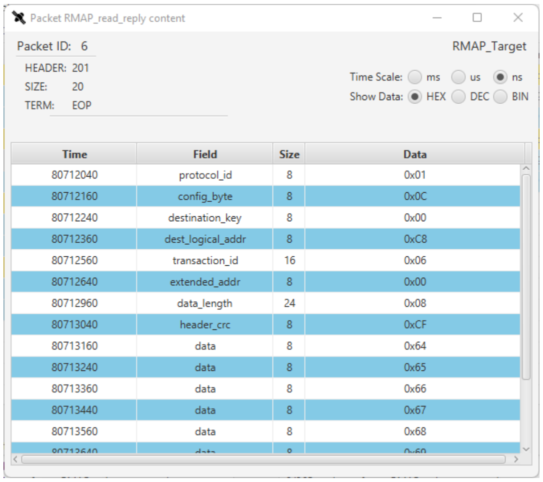
518/1482 packets of type RMAP_read_command
518/1482 packets of type RMAP_write_command
0/1482 packets of type RMAP_read_reply

RMAP_Target - XML parsing info:

0/865 packets of type RMAP_read_command
0/865 packets of type RMAP_write_command
259/865 packets of type RMAP_read_reply

XML file loaded: rmap_packets.xml

Fig. 14. Sniffer GUI visualization of the RMAP packets identified thanks to the SpW PDL.



Packet ID: 6 RMAP_Target

HEADER: 201
SIZE: 20
TERM: EOP

Time Scale: ○ ms ○ us ● ns
Show Data: ● HEX ○ DEC ○ BIN

Time	Field	Size	Data
80712040	protocol_id	8	0x01
80712160	config_byte	8	0x0C
80712240	destination_key	8	0x00
80712360	dest_logical_addr	8	0xC8
80712560	transaction_id	16	0x06
80712640	extended_addr	8	0x00
80712960	data_length	24	0x08
80713040	header_crc	8	0xCF
80713160	data	8	0x64
80713240	data	8	0x65
80713360	data	8	0x66
80713440	data	8	0x67
80713560	data	8	0x68

Fig. 15. Sniffer GUI visualization of an RMAP read command packet.

This test shows how the use of the SpW PDL in the SpW traffic analysis can simplify the work for a SpW user. The user can avoid searching for specific packets by checking their content one by one. The use of a human-readable XML file, hence user-friendly, allows the user to simply identify specific packets in the analysed communication. Moreover, the user can check the structure of the packet and, in this case, check the value of each RMAP-specific field, to find eventual failure in the communication.

In conclusion, the test shows the benefit of the use of the SpW PDL in a SpW application, in this case in the analysis of the communication of a SpW link. Furthermore, the SpW PDL could be applied potentially to any other SpW application which needs to process SpW packets, for example, in the same described test-scenario, the structure of the RMAP packet could have been defined in an XML file and used to generate the data traffic to be sent by the SpW nodes, in an easy and automated way.

Future works on SpaceART [9] have the aim of integrating the SpW PDL in the SpaceART EGSE, precisely for generating SpW traffic with a user-friendly, straightforward, effortless approach. This will provide an instrument to test SpW nodes and also be used along with the SpaceART Sniffer to simulate complex SpW communication test scenarios.

VI. Conclusion

This paper has presented the implementation of the SpaceWire Packet Description Language, an XML-based language for the description of SpW packets, and its integration into the SpaceART SpaceWire Sniffer system.

At first, an introduction about EGSE and SpaceWire has been given, to introduce the need for the design of the SpW PDL. Also the SpW packet format was described, introducing the RMAP protocol, to later describe a test scenario to demonstrate the capabilities of the SpW PDL. Later on, the SpaceART SpaceWire Sniffer has been described. This is a SpW EGSE: a powerful SpW link analyser, capable of storing several minutes of SpW data transmission, for SpW traffic investigation.

The implementation of the SpW PDL is then described, along with its features, showing some examples. After that the SpW PDL has been presented, its integration into the SpaceART SpW Sniffer is mentioned, showing its use for SpW traffic analysis in a simple use-case. Finally, a complete test on an RMAP communication is shown, focusing on the analysis of SpW traffic and RMAP-formatted SpW packet, demonstrating the potential of the SpW PDL and its use in a real SpW test application.

References

[1] S. Parkes and P. Armbruster. Spacewire: a spacecraft onboard network for real-time communications. In *14th IEEE-NPSS Real Time Conference, 2005.*, pages 6–10. IEEE, 2005.

[2] S. Parkes, P. Armbruster, and M. Suess. Spacewire on-board datahanding network. *ESA Bulletin*, 145:34–45, 2011.

[3] S. Parkes, P. Armbrusterand Y. Sheynin, and M. Nomachi. Spacewire missions and architectures. In *60th International Astronautical Congress 2009, IAC 2009*, pages 2963–2970. International Astronautical Federation, 2009.

[4] D. Roberts and S. Parkes. Spacewire missions and applications. In *2010 SpaceWire International Conference*, volume 13, pages 431–436, 2010.

[5] European Cooperation for Space Standardization (ECSS). Spacewire–links, nodes, routers and networks. *ECSS-E-ST-50-12C Rev.1, Noordwijk*, 2019.

[6] European Cooperation for Space Standardization (ECSS). Spacewire protocol identification. *ECSS-E-ST-10-02C Rev.1, Noordwijk*, 2010.

[7] G. Rakow, R. Schnurr, and S. Parkes. Spacewire protocol id: What does it means to you? In *2006 IEEE Aerospace Conference*, pages 7–pp. IEEE, 2006.

[8] European Cooperation for Space Standardization (ECSS). Ground systems and operations. *ECSS-E-ST-70C, Noordwijk*, 2019.

[9] A. Marino, A. Leoni, L. Dello Sterpaio, P. Nannipieri, G. Dinelli, G. Benelli, D. Davalle, and L. Fanucci. Spaceart spacewire and spacefibre analyser real-time. In *2020 IEEE 7th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pages 244–248. IEEE, 2020.

[10] R. Ciardi, S. Vagaggini, A. Marino, and L. Fanucci. Design of a test equipment prototype for spacewire data generation and processing in a specific time-constrained test scenario. In *2022 IEEE 9th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pages 593–597, 2022.

[11] R. Ciardi, A. Marino, G. Benelli, and L. Fanucci. Design and development of an embedded architecture for spacewire network sniffer. In *Embedded World Conference*, volume 2022, pages 175–181, 2022.

[12] European Cooperation for Space Standardization (ECSS). Spacewire remote memory access protocol. *ECSS-E-ST-50-52C, Noordwijk*, 2010.

[13] European Cooperation for Space Standardization (ECSS). Spacewire ccsds packet transfer protocol. *ECSS-E-ST-50-53C, Noordwijk*, 2010.

[14] M. Baldi and F. Risso. Using xml for efficient and modular packet processing. In *GLOBECOM'05. IEEE Global Telecommunications Conference, 2005.*, volume 1, pages 6–pp. IEEE, 2005.

Onboard Equipment (Long)

A SpaceWire/SpaceFibre architecture based on ADHA

Robin Franz
Airbus Defence and Space GmbH
Friedrichshafen, Germany
robin.franz@airbus.com

Julian Bozler
Airbus Defence and Space GmbH
Friedrichshafen, Germany
julian.bozler@airbus.com

Michael Stähle
Airbus Defence and Space GmbH
Friedrichshafen, Germany
michael.staehle@airbus.com

Felix Siegle
European Space Agency
Noordwijk, The Netherlands
felix.siegle@esa.int

***Abstract*—Nowadays all traditional satellites are based on electrical architecture interconnecting units containing electronics modules. The space industry understood the benefit of having flexible units that can welcome recurrent and generic modules to be then complemented by specific modules for targeting different applications. However, the standardisation effort necessary for the module compatibility has not been technically and programmatically coordinated among the companies in the space industry. In parallel, the need of real time information created new opportunities for mass production of small satellites, which rapidly adopted the model of CubeSats composed of boards having standardised interfaces. To meet the three technology targets about spacecraft development time reduction, cost efficiency, and finally faster development and adoption of innovative technologies, ESA in cooperation with all European industry has initiated the Advanced Data Handling Architecture (ADHA). ADHA is based on standardised, interchangeable, and interoperable electronics modules with last generation of microelectronics components. The data exchange of the modules is – dependent on the needed data rates - realized with CAN-Bus, SpaceWire and/or SpaceFibre. This paper introduces the concept of ADHA, the standardization approach and finally the architecture and protocol for SpaceWire and SpaceFibre within ADHA.**

Keywords—Standardization, SpaceWire, SpaceFibre, Advanced Data Handling Architecture (ADHA); Data-Handling; On-Board Computer; modular avionics; integrated and modular architecture; standardised, interchangeable, and interoperable electronic modules; CompactPCI-Serial-Space (cPCI-S-S).

I. Introduction

In 2019 ESA started the definition with all European industry of an Advanced Data Handling Architecture (nowadays called ADHA-1) [1][2][3] with the goal to reduce volume, mass, and power of the Data-Handling System.

The identified solution based on units containing standardised, interchangeable, and interoperable electronics modules also aim to procure these modules from different suppliers and to be finally integrated by integrators in ADHA units to target platforms and payload Data Handling applications. Very quickly all industry responded positively and in 2020, 32 companies agreed at an industrial workshop [4] to adopt and use the cPCI-S-S standard [5] for the ADHA modules and the electrical backplane interfacing them.

In September 2021 two parallel activities (called ADHA-2) [6] run by key industrial data handling stakeholders (ADS/TAS/OHB primes, On-Board Computers and Data Handling unit integrators and module suppliers) aims for spacecraft development time reduction, cost efficiency and faster development and adoption of innovative technologies.

II. Concept

The concept is in line with the specifications of Space AVionics Open Interface aRchitecture (SAVOIR) [7] [8][9][10] and the system requirement of ESAs High Priority Candidate Missions (SRD/ECSS/OIRD/PA) [11][12][13] [14].

A. Rack based solution

The boards are no longer hosted in separated boxes but together in racks. This allows to share a common power converter and to include the interfaces between the boards on the backplane. Such a rack oriented and fully redundant architecture is shown in Figure 1.

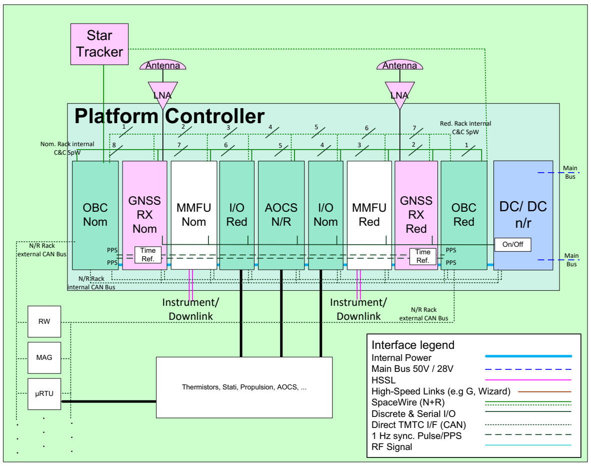


Figure 1: ADHA rack-based architecture

B. Standardisation of boards within the rack

Standardisation of the boards is needed to allow a modular and scalable approach, allowing different suppliers to manufacture boards for such a rack. Furthermore, this allows adapting the CPU performance to the application needs. A common backplane standard was selected by the Compact PCI® Serial Space Standard. The data communication, i.e., the CAN-Bus, the SpaceWire and the high-speed links realised with SpaceFibre, is as well standardized. This will ensure that boards from different suppliers are interoperable and interchangeable.

C. *Command and Control*

Rack internally the boards are connected with SpaceWire to the nominal and redundant OBC. In addition, a rack internal CAN bus will be used as alternative to control the boards. This is beneficial for simple boards which do not exchange many data with the OBC, e.g. I/F-boards. Rack externally the OBC provides SpaceWire Connections. This will be used for command and control for equipment with higher data rates.

For command and control of low data rates equipment a CAN bus is selected for interconnection e.g., for µRTUs.

µRTUs are easy to accommodate in the satellite close to the place where they are needed reducing the harness mass. With their high granularity and configurability µRTU can be added in a late design phase. This ensures that the number of unused interfaces is reduced.

ADHA-µRTU will be built on the same standardization principals but with a form factor of 3U.

The ADHA rack and the corresponding µRTU is shown in Figure 2.

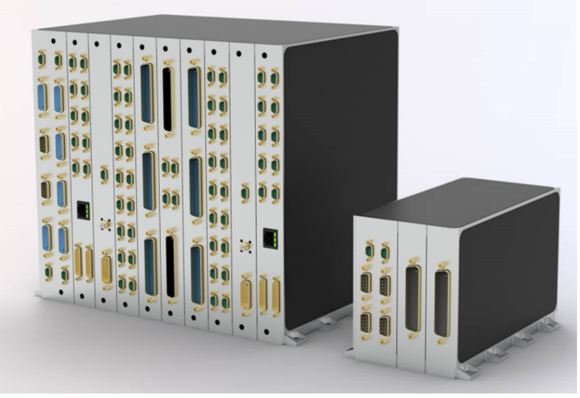
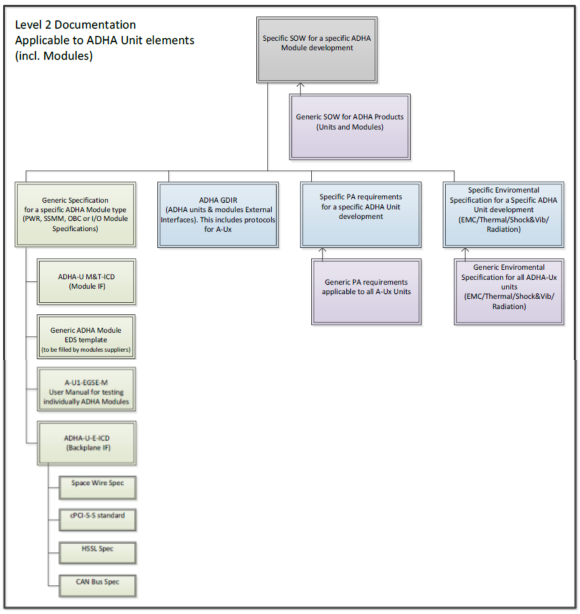


Figure 2: ADHA rack (6U) left and ADHA µRTU (3U) right

III. Standardization approach

The aim of the standardization is to reach an interoperable, interchangeable, and modular system to increase the re-use of units, modules, software, and the test system including test-specifications, -procedures and –scripts.

ADHA-2 will generate a set of public and standardised (ITT) documents and specifications that will federate the future development and procurement of ADHA products (units, modules, EGSEs, mechanical housings and backplanes) by different European companies under the ADHA program. The data package includes generic specifications which can be made application specific by specific or jacket specifications. The data package will include a Statement of Word, Interface Specifications including protocol definitions – among others for SpaceWire and SpaceFibre, Environment Specifications, PA requirements and the EICD as shown in Figure 3.



Level 2 Documentation
Applicable to ADHA Unit elements
(incl. Modules)
Specific SOW for a specific ADHA Module development
Generic SOW for ADHA Products (Units and Modules)
Generic Specification for a specific ADHA Module type (PWR, SSMM, OBC or I/O Module Specifications)
ADHA GDIR (ADHA units & modules External Interfaces). This includes protocols for A-Ux
Specific PA requirements for a specific ADHA Unit development
Specific Enviromental Specification for a Specific ADHA Unit development (EMC/Thermal/Shock&Vib/ Radiation)
ADHA-U M&T-ICD (Module IF)
Generic PA requirements applicable to all A-Ux Units
Generic Enviromental Specification for all ADHA-Ux units (EMC/Thermal/Shock&Vib/ Radiation)
Generic ADHA Module EDS template (to be filled by modules suppliers)
A-U1-EGSE-M User Manual for testing individually ADHA Modules
ADHA-U-E-ICD (Backplane IF)
Space Wire Spec
cPCI-S-S standard
HSSL Spec
CAN Bus Spec

Figure 3: Typical data package to procure a module

A. *SpaceWire and High-Speed Serial Link Architecture*

1) *SpaceWire*

The network architecture is based on a two-router dual star topology physically routed on PCBS tracks over the backplane of the ADHA Unit. The SpaceWire dual star is illustrated in Figure 4.

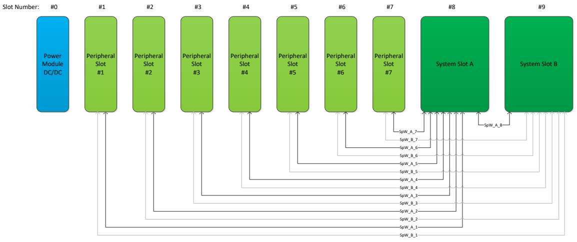


Figure 4: ADHA-internal SpaceWire Architecture

The two System Controller Slots may be used in hot or cold redundancy. The data rate is targeted at 100Mbps. The corresponding Data and Strobe skew and jitter budget is derived. A major design goal is to achieve a single point failure free network which does not require dedicated driver, receiver, or router devices on the ADHA Boards. A direct physical connection of e.g., an FPGA to the SpaceWire network is envisaged to reduce the footprint, simplify layout and routing as well as reduce manufacturing and process qualification costs. This shall be achieved by elaborate requirements on the power supply.

The system slot module shall contain a SpaceWire-router and in addition up to eight external SpaceWire Links to connect other equipment like Star Trackers, ICUs and other ADHA units.

2) *High Speed Serial Links*

A full mesh network for high-speed links is integrated in the ADHA unit, as shown in Figure 5. This allows a high-speed data exchange between all the modules.

Each line consists of two transmitting differential pairs and two receiving differential pairs, as shown in Figure 6.

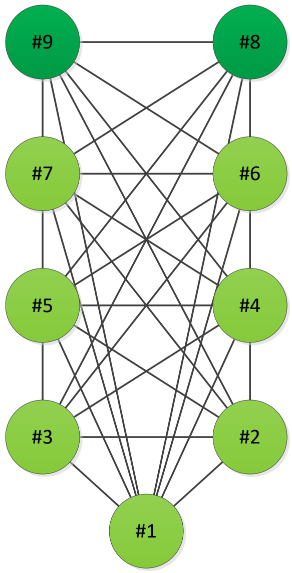


Figure 5: Full mesh topology for high-speed serial links

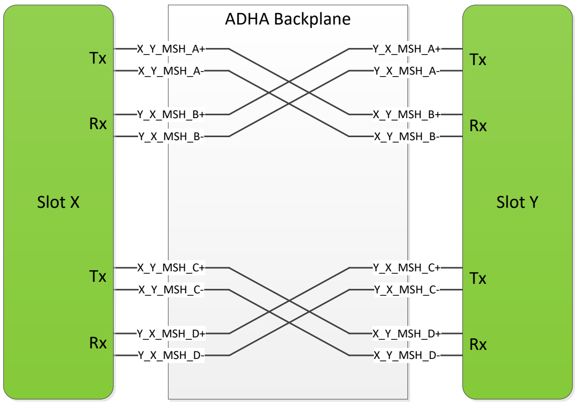


Figure 6: High speed serial link implementation with two transmitting and two receiving differential pairs

B. SpaceFibre Architecture

Conceptually, the ADHA backplane provides a full mesh of two-lane high-speed link connections between all modules in a unit. For the upcoming Engineering Model (EM) of the ADHA unit, SpaceFibre has been chosen as the high-speed link protocol. Rather than making use of the full mesh, the traffic will be routed by a SpaceFibre router, which is placed at the center of a star topology. In terms of high-speed communication, the Mass Memory Unit (MMU) module is the central point that receives data from various sources (instruments, other ADHA modules) and transmits data to various data sinks (payload data transmitter, other ADHA modules) and it is therefore a natural choice to integrate the SpaceFibre router with the memory function on the MMU module.

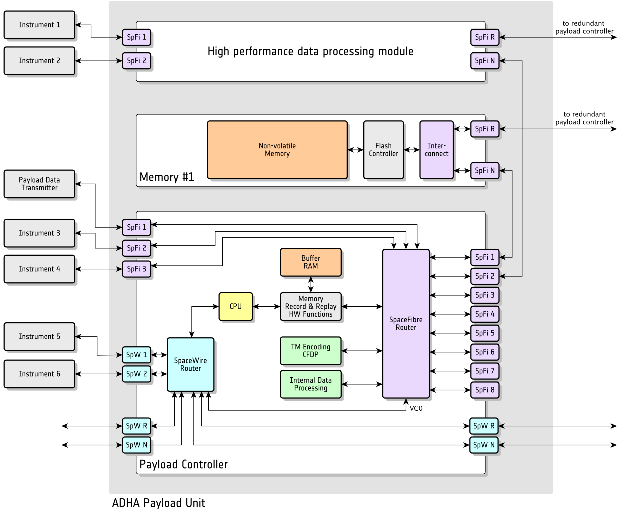


Figure 7: Example of a ADHA MMU module

A potential architecture of the MMU is shown in Figure 7. It consists of a Payload Controller module that comprises several SpaceFibre and SpaceWire interfaces on the front-panel. These interfaces are used to receive data either directly from instruments or from other ADHA modules, which might act as front-end or data processing modules. The front-panel interfaces can also be used to replay data from the memory to the payload data transmitter for downlink or to other ADHA modules for further data processing.

Likewise, the Payload Controller module comprises SpaceWire and SpaceFibre interfaces on the backplane: The nominal and redundant SpaceWire interfaces are connected to the system controller slot (hosting the OBC module in the EM) and the SpaceFibre interfaces are connected to all other peripheral module slots. These interfaces are treated just like the front-panel interfaces, that is, data can be recorded from them, and data can be replayed to them.

The Payload Controller module includes all other functionality of a typical MMU, e.g., a CPU for the file management, hardware functions for filtering, storing, and retrieving data, functions for the telemetry and CFDP encoding, and fast buffer memory. A novelty of this approach, however, is the separation of the flash memory, which is hosted on one or more additional ADHA modules. Data is transferred to/from these memory modules over the backplane via SpaceFibre using a low-level block access protocol.

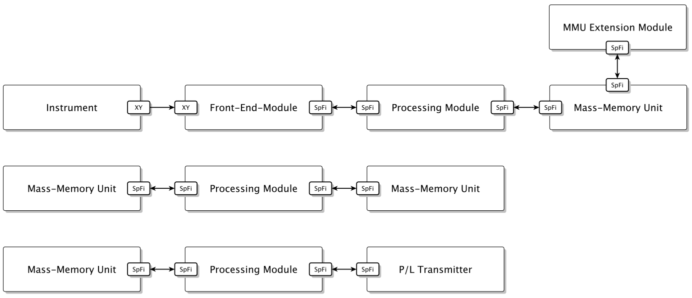


Figure 8: Different data processing chains

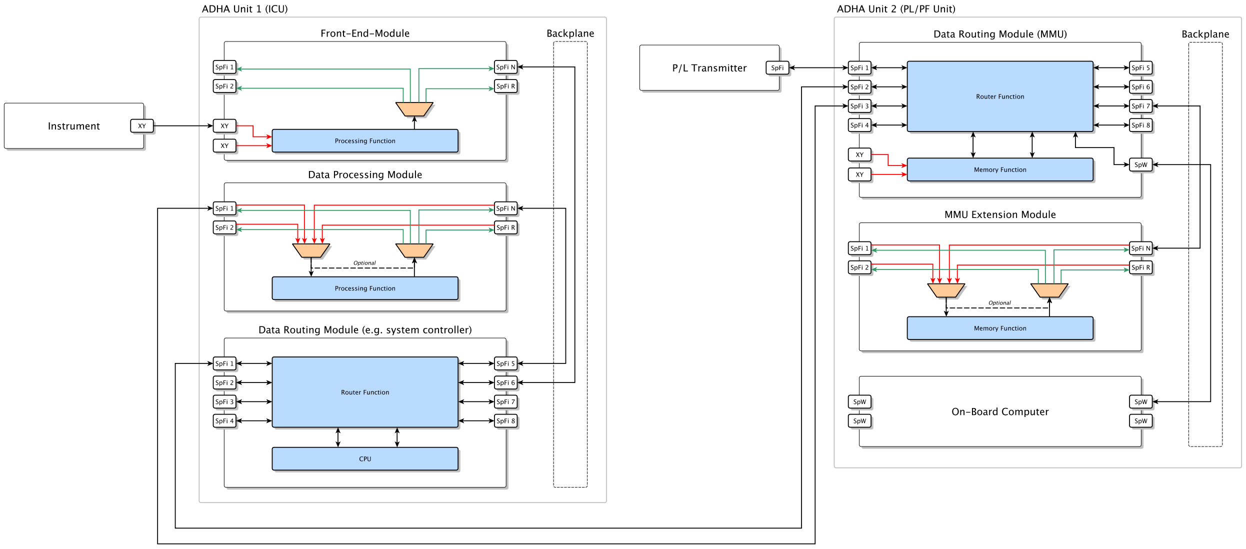


Figure 9: Example of an ADHA system with two units

This concept allows the implementation of payload data processing chains in a very flexible way, interconnecting several ADHA modules that can even be distributed over several ADHA units. Three use cases are shown in Figure 8:

- The first chain is a pre-processing chain, that receives instrument data via a custom data link (e.g., LVDS) into a front-end module, which might do some basic data processing before reformatting the data into SpaceFibre packets to be transmitted further to a processing module. The processing module might do some heavier data processing on the fly, before forwarding the data for storage in the mass memory unit.
- The second chain implements an offline processing approach, where data is replayed from the MMU module to a processing module. Data compression is a typical application example. Then, the data is streamed back to the MMU module for storage into another file.
- The third chain is a post-processing chain, that replays data from the MMU module to a processing module, which processes the data before forwarding it directly to the payload data transmitter. Data encryption is a typical application example.

Since SpaceFibre allows traffic separation via Virtual Channels, a link can simultaneously be used to transmit science data (CCSDS packets) but also command & control traffic (RMAP and CCSDS PUS packets).

To illustrate the mapping of such processing chains onto an actual ADHA system, consider the setup shown in Figure 9 with two ADHA units, one acting as an Instrument Control Unit (ICU) and the other one as a Payload/Platform Unit, combining the On-Board Computer with the MMU.

The ICU unit comprises a front-end module that receives data from an instrument. It is connected via SpaceFibre backplane links to a data routing module. The data routing module is also connected to a data processing module via the internal backplane. Via two external links, the ICU unit is connected to the MMU module of the Payload/Platform Unit. One link connects to the data routing module of the ICU and the other link to the data processing module of the ICU.

- Pre-processing chain: The instrument transmits data to the ICU front-end module. From there, SpaceFibre packets are sent further to the ICU data processing module via the ICU data routing module. After processing, the data is forwarded to the MMU module, placed in the second ADHA unit, for storage. This can either be done through the front-panel interface of the ICU data processing module or through the front-panel interface of the ICU data routing module.
- Offline processing chain: The MMU module replays data via its external SpaceFibre interface to the ICU data processing module. This can either be done via the direct link or indirectly via the ICU data routing module. After data processing, the data is transmitted back to the MMU for data storage into a new file.
- Post-processing chain: Like in the offline processing chain described above, the MMU module sends data to the ICU data processing module, which then sends the processed data back to the second unit. However, this time the data is not stored in memory but is directly forwarded to the payload data transmitter via the embedded SpaceFibre router.

IV. Conclusion

ADHA is an ambitious program to prepare future satellite missions in the domain of On-board Computer Data Handling systems. It is based on a simple, flexible, scalable concept made of standardised, interchangeable, and inter-operable electronics modules.

The need for module standardisation (to produce more rapidly at lower costs) associated to new technologies (modules based on multicore processors, machine learning and artificial intelligence techniques, COTS, etc.) will make ADHA a disruptive approach in the domain of on-board data processing to obtain latest until 2025 highly integrated and scalable units ready to fly.

REFERENCES

[1] Final Report of ADHA-1 ESA Contract 4000124946 with the consortium of RUAG(SE) and Airbus DS (GE), Definition and Roadmap for an advanced Data Handling Architecture for EO Satellites.

[2] Final Report of ADHA-1 ESA Contract 4000124947 with TAS (IT), Definition and Roadmap for an advanced Data Handling Architecture for EO Satellites.

[3] A common ADHA architecture and the used standards, Julian Bozler (ADS), Dario Pascucci (TAS), Jan Johansson (RUAG), ESA Workshop on Avionics, Data, Control and Software Systems ADCSS 2020 presentation.

[4] Advanced Data Handling Architecture & Backplane Dedicated Thematic Workshop Report, J.Bozler (ADS); J.Johansson (RUAG); D.Pascussi (TAS); W.Gasti (ESA).

[5] cPCI-SS https://www.picmg.org/

[6] ADHA-2: Advanced Data Handling Architecture (ADHA) Consolidation, Standardisation and Product Suite Development.

[7] ESTEC, SAVOIR Functional Reference Architecture [ASRA], SAVOIR-TN-001

[8] ESTEC, SAVOIR On-Board Software Reference Architecture [ASRA], SAVOIR-TN-002

[9] ESTEC, SAVOIR generic RTU specication, SAVOIR/12-003/GM

[10] ESTEC, RTU Operability Requirements, TEC-EDD/2013-11/GM

[11] Copernicus High Priority Candidate Missions CO2M Space Segment Requirements Document

[12] Copernicus Expansion tailoring and verification items for ECSS Engineering standards (platform)

[13] Copernicus / Sentinels HPCM OIRD, COP-RS-ESC-FS-6000 issue 1, revision 2.

[14] HPCM Product Assurance and Safety Requirements, Phase B2/C/D/E1

[15] TDE 2021-2022 Work Plan, ESA-TECT-WP-020340, see ESA STAR Publication 1-10601, https://esastar-publication.sso.esa.int/ESATenderActions/details/6948.

[16] ADS ADHA future products and targeted applications, Julian Bozler (ADS), ESA Workshop on Avionics, Data, Control and Software Systems ADCSS 2020 presentation.

[17] TAS ADHA future products and targeted applications, Dario Pascucci (TAS), ESA Workshop on Avionics, Data, Control and Software Systems ADCSS 2020 presentation.

[18] GT1I-304ED - Machine Learning-Based on board Autonomy, Failure Prognostic and Detection.

[19] spaceAPPS - An OPS-SAT Experiment, Hans-Jürgen Herpel (ADS), DASIA 2021

[20] Advanced Data Handling Architecture for Earth Observation Satellites, Julian Bozler (ADS), DASIA 2021

[21] Modular and Interoperable Advanced Data Handling Architecture (ADHA) for Earth Observation (EO) Satellites, Peter Anders (TAS), DASIA 2021

[22] ESA's Technology Strategy, Nov. 2019,ESA/IPC(2018)93.

SpaceFibre Payload Data-Handling Network

Steve Parkes
STAR-Dundee
Dundee, Scotland, UK
steve.parkes@star-dundee.com

Albert Ferrer, Alberto Gonzalez and Marti Farras,
STAR-Barcelona
St Cugat, Barcelona, Catalonia, Spain
enquiries@star-dundee.com

Dave Gibson, Blair Winton, Pete Scott, Chris McClements, Keith Carruthers, Neil Purves, Balint Furdek and Gareth McPherson
STAR-Dundee
Dundee, Scotland, UK
enquiries@star-dundee.com

***Abstract*— The Hi-SIDE project [1] is a European Union project carried out by several leading aerospace organisations from across Europe. It aims to develop satellite data-chain technologies for future Earth observation and Telecommunication systems. Hi-SIDE has made substantial advances in the major elements of the data chain including networking, processing, compression, and downlink transmission to support the next generation of data intensive missions. The data chain elements are interconnected via a SpaceFibre network [2]. This paper introduces SpaceFibre and the Hi-SIDE project and then describes the STAR-Tiger SpaceFibre routing switch which forms the heart of the SpaceFibre network.**

Keywords—SpaceFibre, Payload Data-Handling, Serial Communications, Networks, Hi-SIDE

I. INTRODUCTION

SpaceFibre [2] is the latest generation of SpaceWire [3] network technology for spacecraft on-board data-handling. It runs over electrical or fibre-optic cables, operates at very high data rates, and provides in-built quality of service (QoS), and fault detection, isolation and recovery (FDIR) capabilities. Because of these important characteristics, SpaceFibre was selected for use as the equipment interconnect for the Hi-SIDE project. STAR-Dundee developed the SpaceFibre interfaces for all the elements of the Hi-SIDE demonstrator along with the SpaceFibre routing switch.

The STAR-Tiger SpaceFibre routing switch is the primary element of the payload data-handling network for the Hi-SIDE project, which is used for transferring data at high data-rates between instruments, mass-memory, data compressor, data processor and downlink transmitters. It is also used to provide the control network used by the control computer to control both the network and the equipment attached to the network.

The STAR-Tiger SpaceFibre routing switch is shown in Figure 1 and has the following key features:

- 10 SpaceFibre ports
    - Two quad-lane ports
    - Eight dual-lane ports
    - Lane speed up to 6.25 Gbit/s
    - Port data rate 9.6 Gbit/s dual-lane port and 19.2 Gbit/s quad-lane port
- 2 SpaceWire interfaces for programming STAR-Tiger FPGA
- Spaceflight TRL5/6 level design
- Electronic components are radiation tolerant EM flight parts or industrial/commercial equivalents of flight parts
- Power consumption 14.2W typical at 20 °C, all links running with lanes speeds of 6.25 Gbit/s
- Conduction cooled
- Operating temperature range: -25 to +55 °C
- 108 x 108 x 68 mm (excluding mounting brackets)

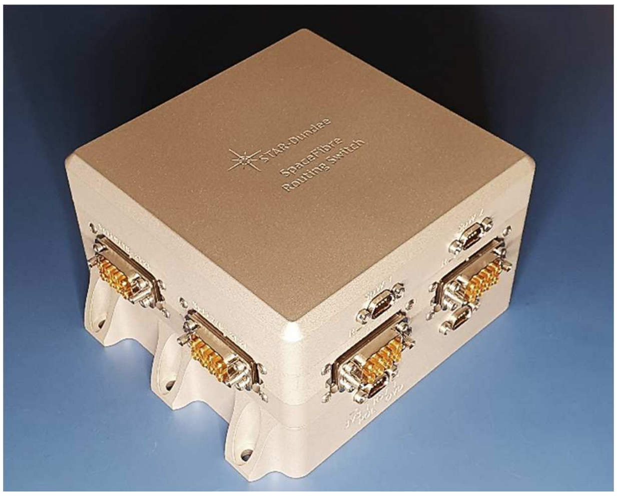


Figure 1: STAR-Tiger SpaceFibre Routing Switch

II. HI-SIDE PROJECT

The Hi-SIDE project has developed critical satellite data-chain technologies for handling and transferring data from instruments to processing and storage elements on-board the spacecraft, and to the downlink transmitters that send data to ground. The Hi-SIDE project culminated in a comprehensive demonstration incorporating all the critical elements of the High Speed Data Chain (HSDC) from instrument to ground-station. A block diagram of the onboard elements of the demonstration system is shown in Figure 2.

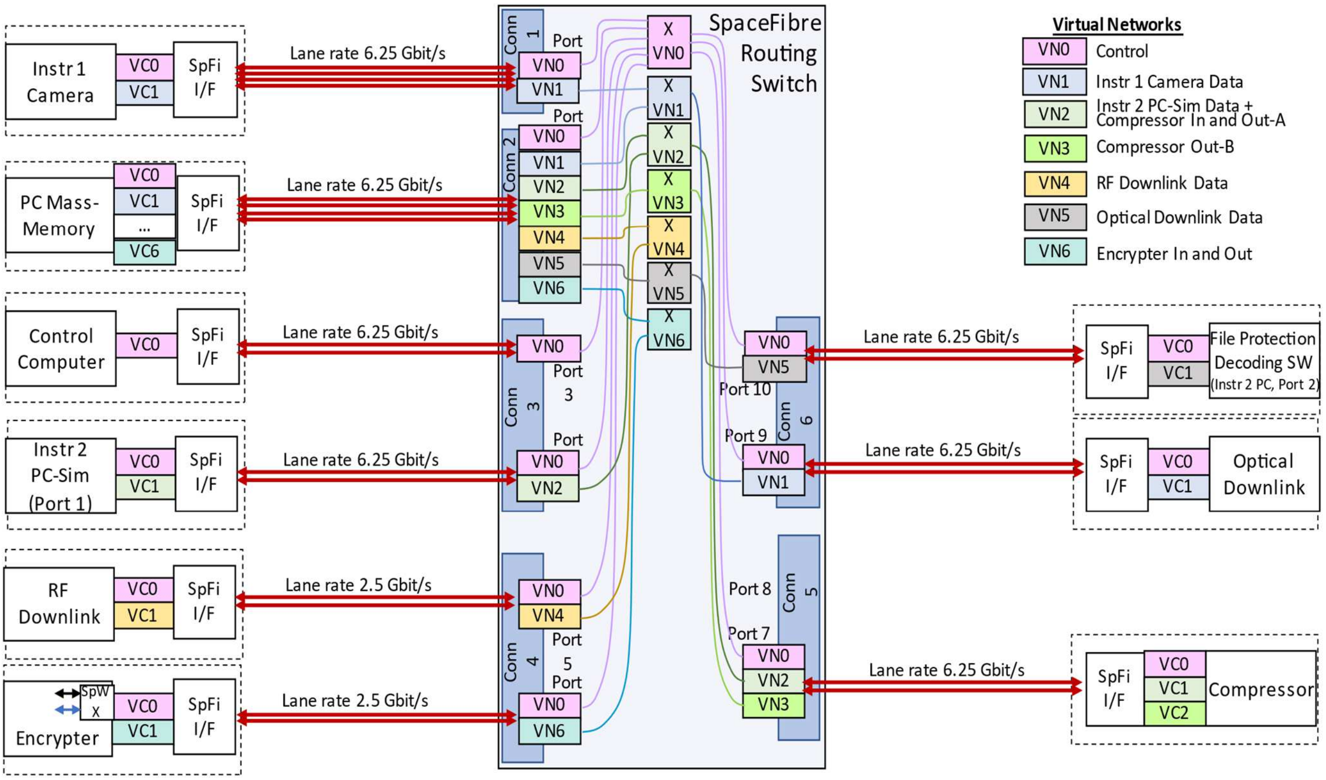


Figure 2: Hi-SIDE Demonstration System Block Diagram

The various elements of the on-board data chain are shown on the left and right sides of the diagram interconnected via SpaceFibre links (red lines) to the routing switch in the middle of the diagram. Each element contains a SpaceFibre interface which connects it to the SpaceFibre network. The interfaces are either quad-lane or dual-lane interfaces, as shown by the number of the lanes in the link to the SpaceFibre routing switch. The interfaces have two or more virtual channels which are mapped by the routing switch to virtual networks. The virtual networks are colour coded and a key to the type of traffic they handle is shown in Figure 2. The configured virtual networks are also illustrated inside the SpaceFibre routing switch (X = virtual network switch).

The HSDC demonstration network includes the following elements:

- STAR-Tiger routing switch connected to all elements via SpaceFibre links.
- Instrument 1 (SpaceFibre camera) connected to STAR-Tiger Port 1, which provides real-time image data at around 4.6 Gbit/s.
- PC-Based Mass-Memory [4] connected to STAR-Tiger Port 2, which stores data from the instruments, passes data to and from data processor/compressor, and sends compressed, encrypted data to the RF or optical downlink.
- Control Computer [4] connected to STAR-Tiger Port 3, which configures, controls and monitors the SpaceFibre network and the equipment connected to the network.
- Instrument 2 (simulator) connected to STAR-Tiger Port 4, which provides hyperspectral data at a data rate of around 9 Gbit/s.
- Radio Frequency (RF) downlink [5] connected to STAR-Tiger Port 5.
- High-Performance Data-Processor (HPDP) [6] connected to STAR-Tiger Port 6, which is programmed to perform data encryption.
- Data Compressor [7] connected to STAR-Tiger Port 7, which is performing CCSDS 123.0-B-2 Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression.
- Image Viewer (simulating the optical downlink [8]) connected to STAR-Tiger Port 9.
- File Protection Scheme (FPS) Decoder [9] connected to STAR-Tiger Port 10.

The SpaceFibre virtual networks separate different types of traffic on the network so that one type cannot interfere with another type. The virtual networks in the demonstration system are used as follows:

- VN 0 is used for network and equipment management and connects to all of the elements. The Control Computer uses VN 0 to send Remote Memory Access Protocol (RMAP) [10] commands to the elements to read or write to registers, and the elements return the corresponding RMAP replies to the Control Computer.
- VN 1 is used by the SpaceFibre Camera to send images either directly to the Image Viewer or for

storage in the Mass-Memory before the Mass-Memory plays back the images to the Image Viewer.

- VN 2 is used by the Instrument Simulator to send data for storage in the Mass-Memory. It is also used by the Compressor to send one half of the compressed files for storage in the Mass-Memory.
- VN 3 is used by the Compressor to send the other half of the compressed files for storage in the Mass-Memory.
- VN 4 is used by the Mass-Memory to send files to the RF Downlink.
- VN 5 is used by the Mass-Memory to send protected files to the FPS Decoder.
- VN 6 is used by the Mass-Memory to send files to the HPDP for encryption, and by the HPDP to send the encrypted files to the Mass-Memory for storage.

A photograph of the integrated demonstration system is shown in Figure 3.

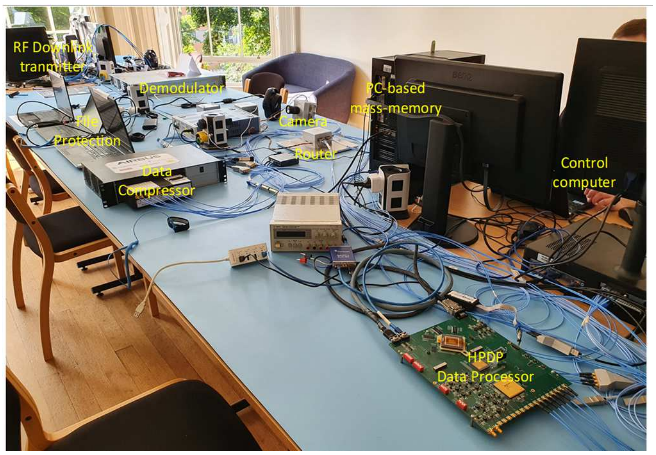
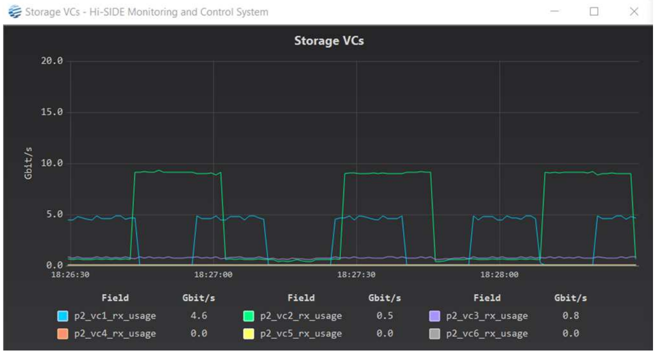


Figure 3: Photograph of the Integrated Hi-SIDE Demonstration System

The control computer was able to monitor and display the data rates of the traffic flowing through each virtual network during the demonstration. Figure 4 shows an example of traffic going to the mass-memory for storage. The chart shows data rates plotted over time for the VCs going into the Mass-Memory used to store data. In this diagram, the following data flows are shown:

- VC 1 (blue line): SpaceFibre Camera sending 8 GB of images to the Mass-Memory for storage.
- VC 2 (green line): Instrument 2 sending 16 GB of data to the Mass-Memory for storage and Data Compressor sending one half of the compressed data to the Mass-Memory for storage.
- VC 3 (purple line): Data Compressor sending the other half of the compressed data to the Mass-Memory for storage.

When these operations overlap, the total data rate of traffic being stored simultaneously in the Mass-Memory is around 14 Gbit/s (the SpaceFibre Camera is approximately 4.5 Gbit/s, Instrument 2 is approximately 9 Gbit/s and the Data Compressor is approximately 0.5 Gbit/s for each of the two compressed streams).


Storage VCs - Hi-SIDE Monitoring and Control System

Field	Gbit/s	Field	Gbit/s	Field	Gbit/s
🟦 p2_vc1_rx_usage	4.6	🟩 p2_vc2_rx_usage	0.5	🟪 p2_vc3_rx_usage	0.8
🟧 p2_vc4_rx_usage	0.0	🟨 p2_vc5_rx_usage	0.0	⬜ p2_vc6_rx_usage	0.0

Figure 4: Example of Monitored Network Traffic to the Mass-Memory

Further information on the Hi-SIDE demonstration system is available in [4].

III. STAR-TIGER SPACEFIBRE ROUTING SWITCH

The on-board network is formed by the STAR-Tiger routing switch, the SpaceFibre cable assemblies and the SpaceFibre interfaces in each element. In this section the STAR-Tiger routing switch is described. The design of STAR-Tiger unit, the router FPGA design, and the functional and environmental testing of STAR-Tiger are outlined.

A. STAR-Tiger Design

The STAR-Tiger SpaceFibre routing switch unit design is illustrated in Figure 5.

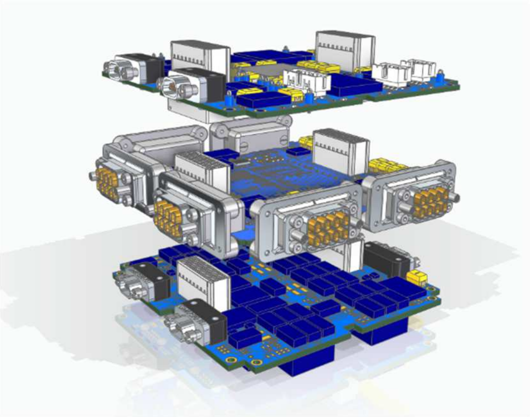


Figure 5: STAR-Tiger Routing Switch Boards

STAR-Tiger comprises three circuit boards:

- A power supply board (bottom) which provides nominal and redundant power input selection and delivers the five main power rails to the FPGA. TI radiation tolerant power supply components are used. Other power rails are supplied by regulators on the other two boards
- An FPGA board (middle) containing the Xilinx KU060 FPGA. An industrial grade FPGA was used. The PCB footprint accommodates either the commercial/industrial part or the radiation tolerant part. The FPGA is surrounded by six Elara connectors which carry the electrical SpaceFibre signals. Each connector provides four lanes of SpaceFibre.
- A configuration and scrubbing board (top). Configuration is from EEPROM or via a SpaceWire interface. The EEPROM can be programmed over SpaceWire.

These three boards are shown in Figure 6 to Figure 8 in various stages of integration with the STAR-Tiger housing. The complete STAR-Tiger unit is shown in Figure 1.

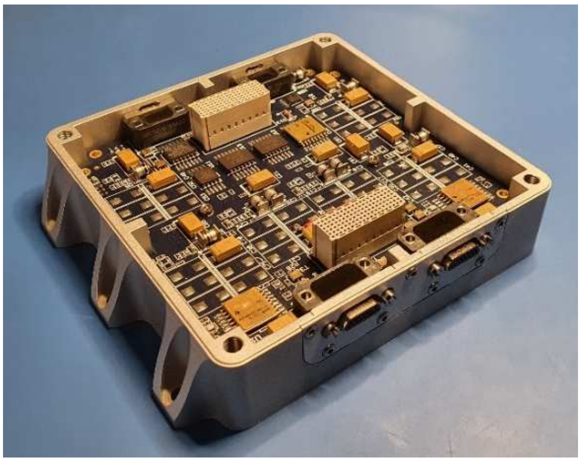


Figure 6: STAR-Tiger Power Supply Board in Housing

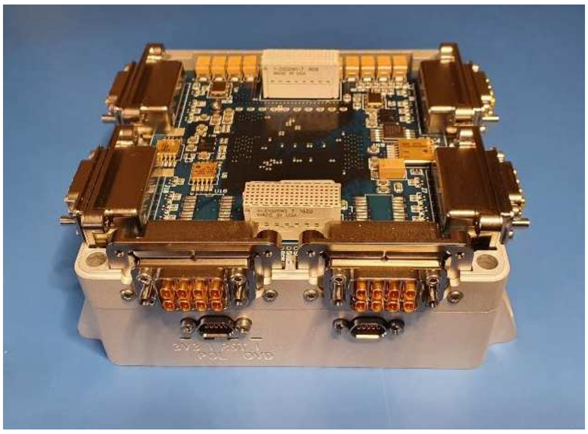


Figure 7: STAR-Tiger FPGA Board in Housing

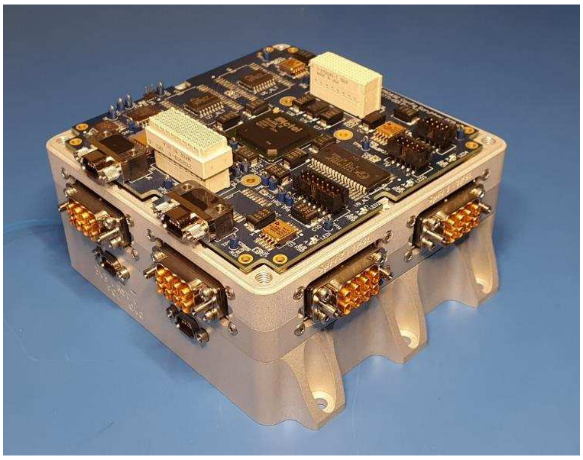


Figure 8: STAR-Tiger Configuration Board in Housing

The SpaceFibre camera used in the Hi-SIDE demonstration was produced by adding an image sensor board to the top of the configuration board which transforms the STAR-Tiger into a high data-rate image sensor.

B. SpaceFibre Routing Switch FPGA Design

A block diagram of the STAR-Tiger SpaceFibre routing switch FPGA is shown in Figure 9. The SpaceFibre routing switch FPGA code was developed by STAR-Barcelona.

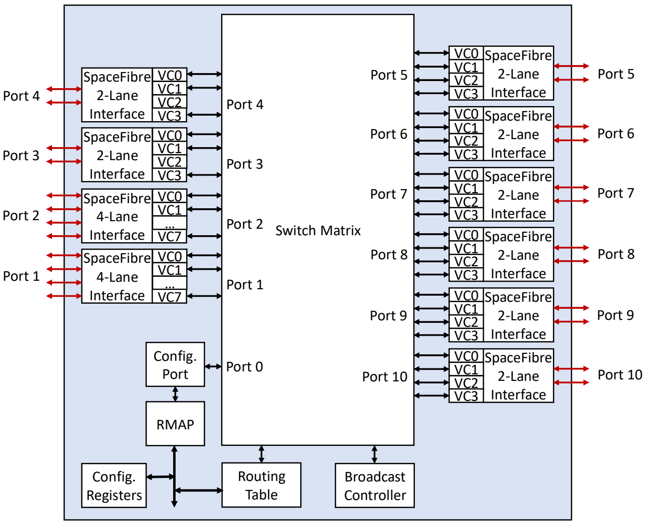


Figure 9: Block Diagram of Routing Switch FPGA

The SpaceFibre routing switch FPGA contains the following:

- A routing switch matrix with ten SpaceFibre ports and an internal configuration port.
- Two quad-lane SpaceFibre ports (ports 1-2) with eight virtual channels each.
- Eight dual-lane SpaceFibre ports (ports 3-10) with four virtual channels each.
- An RMAP configuration port (port 0) which accesses the SpaceFibre router configuration, control and status registers.
- A routing table which is configured over the configuration port and which determines the logical address to output port-number mapping.
- A broadcast controller which broadcasts broadcast-messages on each of the 256 possible broadcast channels. The broadcast controller also provides the time-slot timing for the schedule quality of service.

The placement of each the SpaceFibre ports in the FPGA is illustrated in Figure 10.

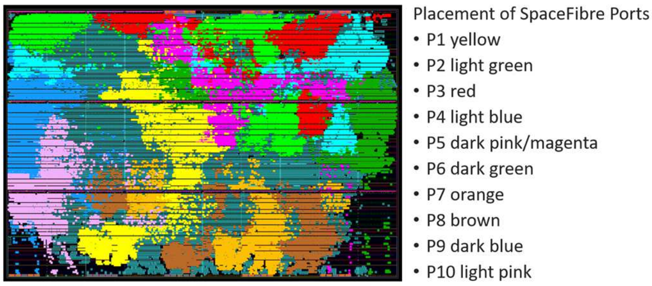


Figure 10: STAR-Tiger FPGA SpaceFibre Port Placement

C. STAR-Tiger Functional Testing

The STAR-Tiger boards were subject to extensive testing during development and integration. Once STAR-Tiger was operational, verification tests were carried out to ensure that the unit performed as required. The test setup used for many of the functional tests is shown in Figure 11.

The STAR-Tiger is in the centre of the photograph, powered by a bench power supply set to 5V with a current limit of 3A. The power lead is connected to the rear of the

STAR-Tiger unit. A cooling fan is used to cool the STAR-Tiger unit. Since the power dissipation of STAR-Tiger is only 14.2W, the cooling fan is set to its minimum setting. The SpaceWire interface device used for programming STAR-Tiger is a SpaceWire Physical Layer Tester (SPLT), with the SpaceWire link to STAR-Tiger being connected to port 3 of the SPLT. Any other SpaceWire interface device could be used. The Python scripts which send command and data to the STAR-Tiger configuration board are run on a laptop PC connected to the SPLT via a USB 2.0 cable. The laptop PC is also connected to a STAR-Ultra PCIe SpaceFibre interface board [11] which is in a Thunderbolt 3 to PCIe unit. The host PC is able to send and receive SpaceFibre packets via the STAR-Ultra PCIe board at the 10 Gbit/s necessary for testing the STAR-Tiger. The STAR-Ultra PCIe is connected to port 2 of STAR-Tiger by a QSFP to Elara cable assembly. To support the testing of all the SpaceFibre ports, port 1 of STAR-Tiger is connected back to itself via a lookback cable and there are cable assemblies between ports 3/4 and ports 5/6, and between ports 7/8 and ports 9/10.

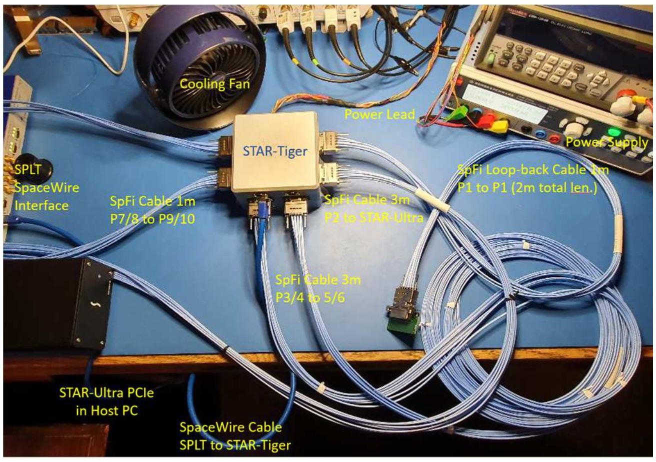


Figure 11: STAR-Tiger SpaceFibre Routing Switch Test Setup Photograph

The STAR-Tiger routing switch was tested using the STAR-Ultra PCIe SpaceFibre interface board which is ECSS-E-ST-11C compliant. The two quad-lane ports were tested first, using the arrangement illustrated in Figure 12.

The lane speed is 6.25 Gbit/s giving a link speed of 25 Gbit/s, which is 20 Gbit/s excluding the 8B10B encoding and 19.2 Gbit/s excluding other protocol overheads for a bi-directional link. 100 Kbyte packets are sent from the STAR-Ultra PCIe to port 2 of the STAR-Tiger router, then out through port 1, looped back to port 1 and finally back out of port 2 to the STAR-Ultra PCIe (see the red path in Figure 12).

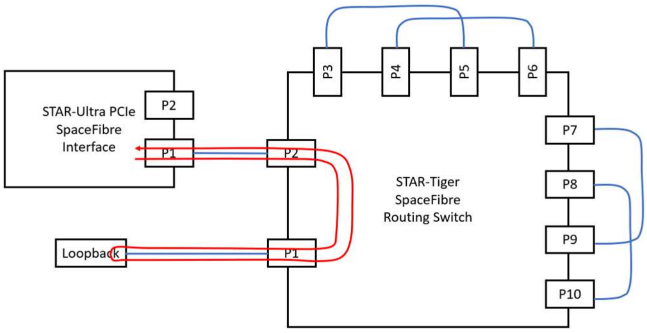
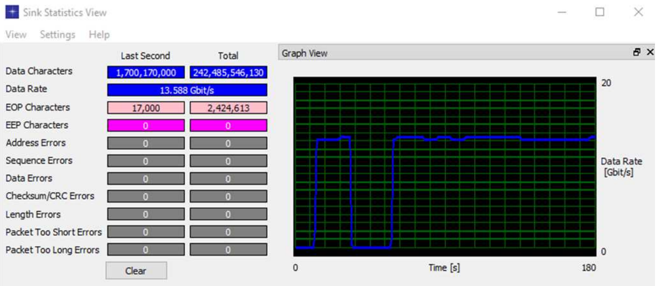


Figure 12: STAR-Tiger SpaceFibre Routing Switch Test Setup

Figure 13 shows a screenshot of the STAR-Ultra PCIe connected to a STAR-Tiger transferring data at a data rate of 13.6 Gbit/s. Data is travelling in both directions of both ports. The source was turned on for around 20s, then switched off for 25s, and then runs continuously. The aggregate data rate being handled by the STAR-Ultra PCIe is 27 Gbit/s (data rate in plus data rate out). The data rate of 13.6 Gbit/s is less than the possible 19.2 Gbit/s because of the time taken by the PC generating the data and performance constraints of the STAR-Ultra PCIe board.



	Last Second	Total
Data Characters	1,700,170,000	242,485,546,130
Data Rate	13.588 Gbit/s	
EOP Characters	17,000	2,424,613
EEP Characters	0	0
Address Errors	0	0
Sequence Errors	0	0
Data Errors	0	0
Checksum/CRC Errors	0	0
Length Errors	0	0
Packet Too Short Errors	0	0
Packet Too Long Errors	0	0

Figure 13: STAR-Ultra PCIe sending and receiving SpaceFibre data at high-speed to/from STAR-Tiger

Tests were then carried out to check all of the links operating together. The test setup is shown in Figure 14. Using path addressing the packets are sent through all the ports of the routing switch and back to the STAR-Ultra PCIe board.

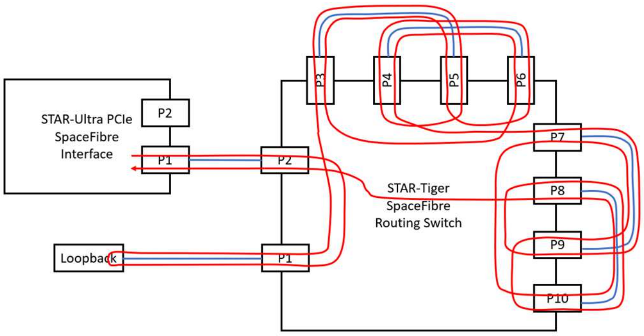
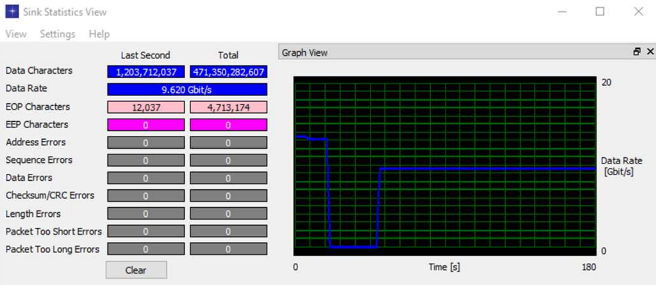


Figure 14: Testing all SpaceFibre ports of STAR-Tiger

The test results are shown in Figure 15. At the start of the trace only the two quad-lane ports were being used, giving a data rate around 13.6 Gbit/s. The path address was then changed to include all the dual-lane ports and the data rate drops to around 9.6 Gbit/s, which is the maximum data-rate that can be supported with two-lanes and a lane speed of 6.25 Gbit/s. Further checks were carried out forming a comprehensive set of verification tests.



	Last Second	Total
Data Characters	1,203,712,037	471,350,282,607
Data Rate	9.620 Gbit/s	
EOP Characters	12,037	4,713,174
EEP Characters	0	0
Address Errors	0	0
Sequence Errors	0	0
Data Errors	0	0
Checksum/CRC Errors	0	0
Length Errors	0	0
Packet Too Short Errors	0	0
Packet Too Long Errors	0	0

Figure 15: STAR-Ultra PCIe exercising all SpaceFibre ports of the STAR-Tiger

D. STAR-Tiger Environmental Testing

With the functional and performance verification tests complete, STAR-Tiger was subject to some environmental testing, covering thermal, vibration and radiated emission tests.

A thermal simulation of STAR-Tiger is shown in Figure 16.

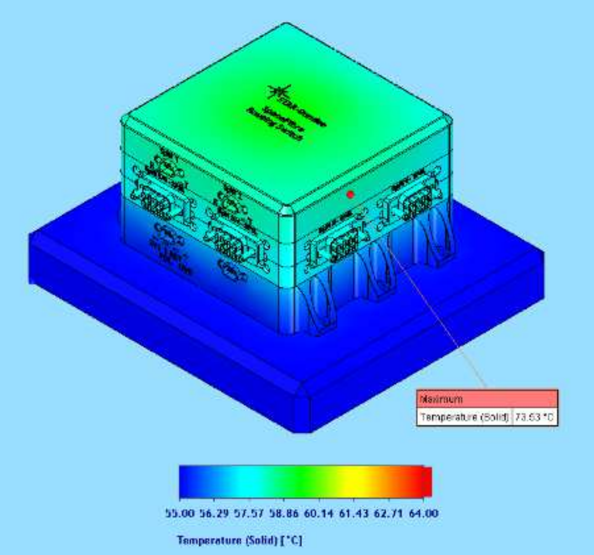


Figure 16: Thermal Simulation of STAR-Tiger (18W)

This particular thermal simulation assumed a worst case power dissipation of 18W, compared to the room temperature power consumption of 14.2W.

The STAR-Tiger unit ready for thermal testing is shown in Figure 17. It is mounted on an aluminium baseplate with heatsinks to keep the baseplate close to the temperature of the thermal chamber. STAR-Tiger is covered with thermal insulation to prevent convection affecting the test results. Thermal cycling was carried out for 15 hours. The results are shown in Figure 18 and correspond to the results of the thermal simulation. There is a temperature drop of around 10°C from the lid of the FPGA to the baseplate. At the end of the temperature test, condensation in the test chamber caused an issue, but STAR-Tiger recovered from this once the condensation cleared.

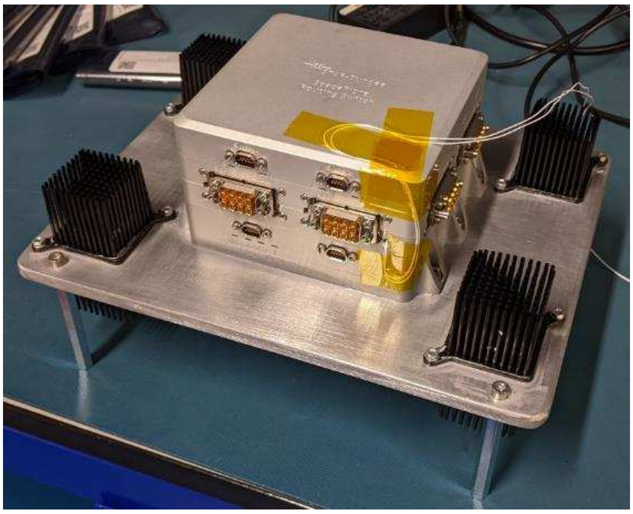


Figure 17: STAR-Tiger Prepared for Thermal Testing

Thermal testing was carried out with a qualification temperature range of -30°C to +60°C for an operational temperature range of -25°C to +55°C.

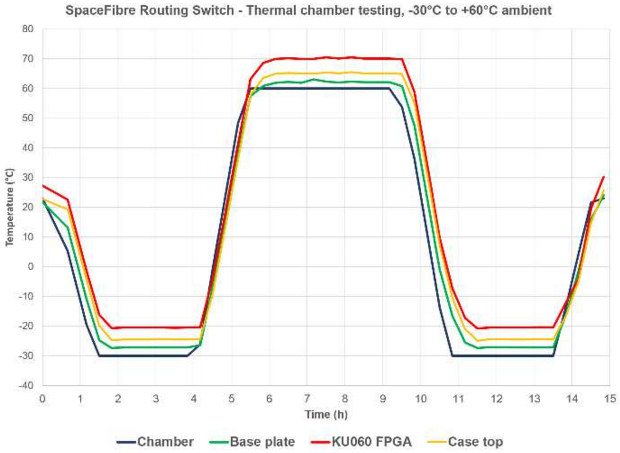


Figure 18: STAR-Tiger Thermal Test Results

Figure 19 shows STAR-Tiger ready for vibration testing. The three axes were tested. For each, an initial scan for resonant peaks was run using a sinewave sweep from 20 Hz to 2 kHz (see Figure 20). Random vibration testing was then carried out for two minutes per axis. A subsequent sinewave scan was then made to see if the resonant peaks had shifted significantly, which would indicate mechanical instability. No significant shift in the frequency and amplitude of the peaks were observed, so the test passed. Note that the limit lines in Figure 20 are for the forcing function, the green line.

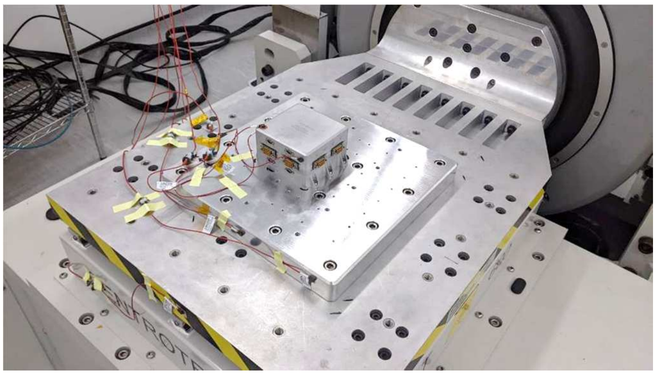


Figure 19: STAR-Tiger Prepared for Vibration Testing

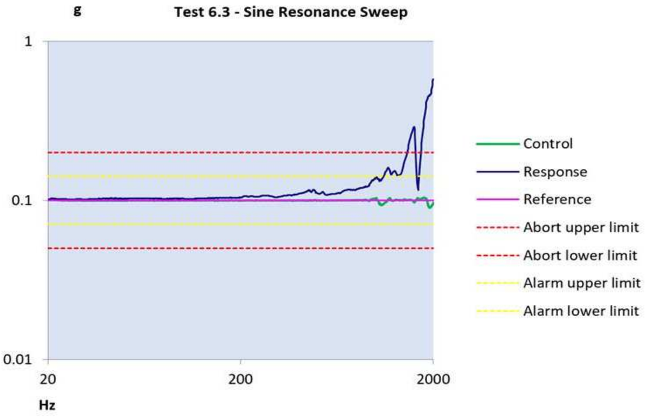


Figure 20: STAR-Tiger Vibration Test Results

EMC radiated emission testing was carried out and those tests were passed (see partial test results in Figure 21). Conducted emission tests have not yet been done.

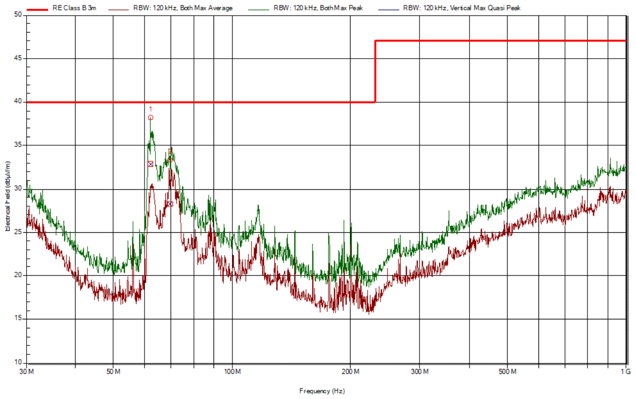


Figure 21: EMC Radiated Emission Testing (30MHz to 1GHz)

IV. Conclusions

The Hi-SIDE project has successfully demonstrated a high-performance data-handling chain for future Earth Observation missions. The STAR-Tiger SpaceFibre Routing Switch forms the heart of the SpaceFibre network that connects the instruments, data-handling and downlink telemetry elements together. STAR-Tiger is capable of data rates up to 19 Gbit/s on its quad-lane ports and 9.6 Gbit/s on it dual-lane ports. STAR-Tiger has been developed with radiation tolerant components to a TRL level of 5/6. It further demonstrates the capabilities of SpaceFibre for future on-board payload processing network.

Acknowledgement

The Hi-SIDE project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 776151.

References

[1] Hi-SIDE Consortium, https://www.hi-side.space/.

[2] ECSS Standard ECSS-E-ST-50-11C, “SpaceFibre – Very High-Speed Serial Link”, Issue 1, European Cooperation for Space Data Standardization, May 2019, available from http://www.ecss.nl.

[3] ECSS Standard ECSS-E-ST-50-12C Rev.1, “SpaceWire – Links, nodes, routers and networks”, European Cooperation for Space Data Standardization, May 2019, available from http://www.ecss.nl.

[4] Gibson, D. et al, “Hi-SIDE: Monitoring, Control and Test Software in a SpaceFibre Network”, International SpaceWire and SpaceFibre Conference, Pisa, Italy, October 17-19th, 2022.

[5] Tesat, ERZIA and Kongsberg, Hi-SIDE Consortium, “High Rate RF Downlink Transmitter”, https://www.hi-side.space/hi-side-rf-data-link

[6] ISD, Hi-SIDE Consortium, “High-Performance Data Processor Payload Unit”, https://www.hi-side.space/copy-of-hi-side-optical-data-link

[7] Airbus UK, NKUA & UoB, Hi-SIDE Consortium, “Data Compression Module”, https://www.hi-side.space/hi-side

[8] DLR, Hi-SIDE Consortium, “Optical Data Link”, https://www.hi-side.space/hi-side-optical-data-link

[9] DLR, Hi-SIDE Consortium, “File Protection Scheme”, https://www.hi-side.space/copy-of-hi-side-rf-data-link

[10] ECSS Standard ECSS-E-ST-50-52C, “SpaceWire – Remote Memory Access Protocol”, Issue 1, European Cooperation for Space Data Standardization, 5 February 2010, available from http://www.ecss.nl.

[11] STAR-Dundee, “STAR-Ultra PCIe”, https://www.star-dundee.com/products/star-ultra-pcie/

Poster Presentations

QUALIFICATION OF ONE AND TWO INTERCONNECTS IN A SPACEWIRE LINK

Derek Schierlmann
Naval Center for Space Technology
Naval Laboratory Code 8243
Washington, D.C., USA
derek.schierlmann@nrl.navy.mil

Michael Long
Naval Center for Space Technology
Naval Laboratory Code 8243
Washington, D.C., USA
michael.long@nrl.navy.mil

Sean Bagnall
Naval Center for Space Technology
Naval Laboratory Code 8243
Washington, D.C., USA
sean.bagnall@nrl.navy.mil

Abstract— **NRL has validated non-standard cabling previously as documented in a paper for the 2008 International SpaceWire conference and applied that knowledge to qualify a new set of non-standard cables. NRL built three four-meter long test cables with zero, one and two series 79 connector pairs inline. These cables were subjected to a test campaign over temperature including time domain reflectometry, insertion loss measurement, bit error rate testing, link rate testing and eye diagram analysis. The test campaign, test equipment and pass/fail criteria are provided as a notional qualification procedure for future non-standard cables. All test results in all test configurations passed at 200Mbps with significant margin. The test results show an expected trend in that the cable with no connections is better than one in-line connection cable which is better than the cable with two inline connections. There was a slight trend in results over temperature in that results at hot temperature are better than test results at ambient temperature which is better than results at cold. The effects are much less impactful than expected in that no measurement deviated more than fifteen percent across all configurations, and all temperatures.**

I. INTRODUCTION

The physical size of our payload and the number of components desiring access to the SpaceWire network created the necessity to investigate alternatives to standard SpaceWire interconnects. By the nature of vehicle integration, it was not always possible or practical to create a point-to-point network connection, as specified in SpaceWire documents. As the concept for the payload layout began to take shape, it became obvious that at least one—if not two—break in a number of SpaceWire links would be required (i.e., the transition from inside the vehicle to outside the vehicle). This need precipitated the development of a qualification process for the inclusion of multiple in-line connectors in a SpaceWire link.

NRL developed a procedure to qualify SpaceWire links and provide confidence that the links will work consistently until the end of the mission. The qualification procedure includes a mix of quantitative and qualitative measurements; the quantitative measurements provided pass/fail criteria and the qualitative measurements provided added understanding of the margins in the system. Both types of assessments were needed. The qualitative pass/fail criteria provided a 'line in the sand' that is needed when test results may force the program to spend additional resources or direct a contractor to do the same. The qualitative criteria provided an understanding of the results, like whether or not the system barely passed or if there was a significant margin.

TABLE 1. SpaceWire Test Steps

Test	Procedure	Pass/Fail Criteria
TDR/TDT, 2 cycles	NCST-TPR-GR104	insertion loss <6dB, impedance =100 +/- 6Ohms
FCT BER, 2 cycles	NCST-TPR-GR105	BER < 1E-9
SPW rate, 2 cycles	NCST-TPR-GR106	>=100Mbps error free
EYE diagram, 1 cycle	NCST-TPR-GR106	Qualitative assessment

SpaceWire test pass/fail criteria consisted of four parts: Link rate testing, Time Domain Reflectometry/Transmission (TDR/TDT) testing, Eye diagrams, and overall assessment. Except for TDR, all testing was to be completed at cold, ambient, and hot temperature. The tests, their controlling procedure documents and pass/fail criteria are shown in Table 1.

The intent of the test was to assess the impact of the required interconnects on data transmission; therefore, a cable was fabricated representing each of the configurations under consideration (one, and two interconnects inline) and a baseline with zero interconnects inline. The cables were fabricated using flight parts and processes by flight certified technicians and were nearly identical cop(ies) of the cables that will fly on the mission. The length of the cables was chosen to represent the worst-case noise situation. The reflections caused by the impedance mismatch of the interconnects are assumed to be the dominant noise contributor, which means the shortest cable is assumed to be the worst case. Thus, all the cables were built to match the shortest cable length expected on our fight SpaceWire cables. The intermediate connector in our one and two interconnect test cables is a 15 position (D-15 contact arrangement) Glenair Series 79 Micro-Crimp connectors place in the center of the cable. The test cables followed the same pin assignment geometry as the flight cable, with the only difference being the addition of two ground pins in the center of the connector compared to our 13-pin flight design.

II. Results

TABLE 2. Test Results

Test	*Test Results Summary*	*Comparison Notes*
FCT BER, 2 cycles	Pass. Successful transmission in both directions at 10, 20, 25, 30, 40, 50, 60, 70, 90, 100,120, 140, 160, 180, and 200 Mbps.	No variations seen. Results remained consistent and passing over all three temperatures plateaus on both thermal cycles
SPW rate, 2 cycles	Pass. Successful transmission in both directions at 100, 125, 150, and 200 Mbps	No variations seen. Results remained consistent and passing over all three temperatures plateaus on both thermal cycles
TDT (insertion loss)	Pass. Maximum insertion loss measurements were considerably below pass/fail limits. maximum variation in S21 at 25MHz where the data varied by +/-5%	A general trend that UUT3 (two interconnects) was most lossy and UUT1 (no interconnects) was least lossy.
TDR (impedance)	Pass. All pairs were in family within each cable. Average impedance varied 5 Ohms (105-110), within spec.	There is no discernable trend seen in the average impedance results. Comparing the UUT3 results to UUT1 shows that the impedance discontinuities at 0 and 2 (Figure 1) are less than at 1 and 3, respectively, which suggests the impedance disconnect of the series 79 is less severe than that caused by the micro-D 9 in the SpaceWire specification [1]
EYE diagram, 1 cycle	Pass. SpaceWire Eye taken diagrams at 200Mbps with all results have monotonic edges and show significant margin against the eye mask.	Eye diagram waveforms of UUT2 and UUT3 show additional reflections in the results that is not present in UUT1; the additional reflection is strongest in UUT3 at ambient and the reflection separates more in UUT2 than UUT3, as if UUT3 is 'nosier' but the effect is not significant.

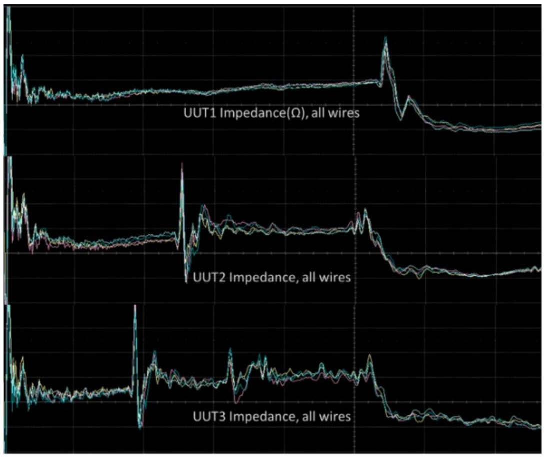


Figure 1. TDR impedance Traces of all Three UUTs (20Ohms/division)

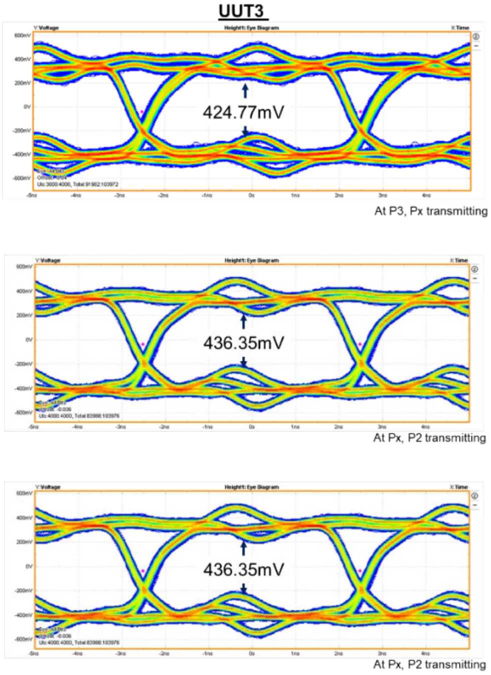


Figure 2: Test Results – Eye Diagram at 200 Mbps, Hot Temp. Results at Ambient and Cold Temperatures are Similar

When tested, all cables passed all test in all configurations. Variations and trends were discernable in only some of the test results as summarized in the results Table 2, TDR traces and Eye diagrams (Fig. 1 and Fig. 2). Overall, results show that the effects of the additional interconnects are un-noticeable is SpaceWire link rate and BER testing and are only discernable after detailed and targeted testing and data analysis. Even then, the interconnects change the detailed data results by no more than 15%. The resulting data trends as expected in that as interconnects are added, data transmission is negatively impacted. Also, the trend in results over temperature is as expected: Hot cables have more resistance that cause more loss and reduce impact of any reflections caused by impedance mismatches. The one and two interconnect cables pass all program requirements and met the SpaceWire specification—with the exception of the connector impedance discontinuity. The test results conclusively show that one and two interconnect cables can be used on the program. These results can be extended safely to the SpaceWire limit of 10m due to the ample margin that these cables demonstrate. But these results would not hold for a cabling configuration that adds significant loss to the system.

III. Conclusions

Comparing this effort to our previous work qualifying nonstandard cables, improvements can be seen in test equipment, the SpaceWire specification, and the series 79 connector. Test equipment is more readily available and easier to apply. The Teledyne/LaCroy SPARQ 3008E Signal Integrity Network Analyzer is a more robust and user-friendly tool to extract impedance and insertion loss than the equipment used previously. Likewise, the Tektronix DSA 70604 Digital Serial Analyzer with DPOJET software significantly automated the capture and analysis of eye diagrams. Both the versions of the Star-Dundee SpaceWire-USB brick were simple to use and we benefited from increased availability of them as NRL has procured more over the years. We also had significantly improved SMA/TDR to micro-D9/UUT test boards that were fully developed under a previous program. The specification's inclusion of an eye mask and insertion loss specifications were also quite helpful in developing the pass/fail criteria.

Finally, when comparing TDR output traces in this paper to those from 2007, the series 79 connector used in this paper appears to have a smaller impedance discontinuity /than the 38999 Series II connector [1]. However, a direct comparison is difficult as the test results are taken by different test equipment and under different conditions. This paper recommends the series 79 connector 15-position connector as a good choice as an intermediate connector and provides Table 1 as recommended qualification procedure. The NRL procedure documents referenced in the table can be provided to limited distribution upon request; other entities can contact the author for a synopsis of the steps.

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA).

[1] D. Schierlmann, P. Jaffe, "SpaceWire Cabling in an Operationally Responsive Space Environment," Proceedings of the International SpaceWire Conference 2007.

[2] D. Schierlmann, E. Rossland, P. Jaffe, "Lessons Learned From Implementing Non Standard Spacewire Cabling For TACSAT-4," Proceedings of the International SpaceWire Conference 2008.

[3] P. Jaffe, G. Clifford, J. Summers, "SpaceWire for Operationally Responsive Space as part of TacSat-4," Proceedings of the International SpaceWire Conference 2007.

SpacePHYre: Magnetically Isolated SpaceFibre Links using Gigabit Ethernet PHYs

Matthew Rowlings, Michael Walshe
Thales Alenia Space UK
Bristol, United Kingdom
matthew.rowlings@thalesaleniaspace.com

Martin Trefzer, Mark Post
Department of Electronic Engineering
University of York
York, United Kingdom
martin.trefzer@york.ac.uk

***Abstract*—SpacePHYre is a new on-board communications CODEC that is compatible with SpaceFibre and SpaceWire networks. SpacePHYre uses magnetically isolated Gigabit Ethernet PHYs at the physical layer to provide a higher degree of electrical isolation between equipment. Deterministic and reliable communication is supported by providing SpaceFibre Virtual Channels to the user, including the Quality of Service mechanisms, as well as a frame retry mechanism based on Space Fibre. In addition, the magnetic isolation allows power delivery over the data cable to be supported, resulting in harness mass savings as dedicated power cabling to the equipment is no longer required. A prototype SpacePHYre interface supporting power delivery based on terrestrial Power-over-Ethernet standards is presented.**

I. Introduction

The simplicity of SpaceWire networking layer allows avionics networks to be created easily and is highly suitable for FPGA implementation or for use in processor-less applications. SpaceFibre improves on SpaceWire networking by supporting reliable transmission over a link and by adding deterministic data delivery through virtual networks and time slot arbitration provided by Quality-of-Service mechanisms(QoS) built into the virtual channel layer. Despite these new capabilities, a SpaceFibre CODEC is still suitable for FPGA implementation [1].

As well as increasing the link speed, the electrical physical layer of SpaceFibre improves on the signal integrity of SpaceWire by offering AC coupling to mitigate the effects of differing ground potentials and by providing protection to the drivers and receivers should a DC transient fault be present on the cable [2]. However, each end of the line is still coupled to a local ground via the discharge resistors and so this will eventually limit the length of the copper cable or maximum transient fault that can be tolerated between two SpaceFibre devices. The AC coupling also does not fully isolate SpaceFibre devices from power supply induced transients that may occur when high-power devices are switched on or off. The fibre optic cables and drivers for SpaceFibre solve this issue but are expensive and difficult to integrate and handle.

The isolation can be improved by moving to magnetic isolation using a transformer between the equipment and the cable, as used in several robust interfaces such as MIL1553 and 10/100/1000BASE-T Ethernet interfaces. Indeed, 1000BASE-T Gigabit Ethernet PHYs combine a suitable combination of high galvanic isolation and a reasonable throughput of 1 Gigabit per second. Space-qualified Gigabit PHY devices are becoming available on the market (Microchip VSC8541RT [3] and Texas Instruments DP83561 [4]), due to their use in Ethernet-based on-board protocols such as Time Triggered Ethernet. Ethernet-based avionics has been used in aerospace applications such as fly-by-wire systems. This technology offers an enhanced full Ethernet stack with extensions for guaranteed delivery of time critical data. However, supporting the full Ethernet stack results in lots of extra features available that are not applicable to spacecraft avionics and as a consequence such a avionics solution requires its own ASIC per interface [5] as well as the physical layer PHYs and magnetics that provide the Ethernet physical layer and the electrical isolation, increasing power consumption and PCB space required for the interconnect compared to an embedded SpaceFibre CODEC with external SERDES IC.

In this paper we propose an alternative 1 Gbit/s physical layer for SpaceFibre using the 1000BASE-T Gigabit physical layer. The magnetic isolation offered by 1000BASE-T provides maximal decoupling of the cable from the interfaces, offering a high level of immunity to power supply induced transients. The use of magnetic isolation also allows the capability of power delivery over the communication cables. The codec implements the SpaceFibre retry mechanism, Quality of Service (QoS) primitives, Broadcast and Virtual Channel interface to allow compliance with SpaceFibre and SpaceWire at the network layer.

The magnetic isolation and Ethernet PHY technology brings several other advantages to on-board avionics networks:

1) ***Cable length:*** The magnetically coupled on-board communication network will be able to support cable lengths of up to 100m, allowing easier integration into larger spacecraft, easier integration with EGSE and new application areas such as space launch systems.
2) ***Power delivery:*** The magnetically coupled on-board communication network will be able to power networked devices (e.g. star trackers, sensors, small instruments) by delivering the power over the same cable as the communication as is done terrestrially with Power over Ethernet (PoE). This removes the need for separate

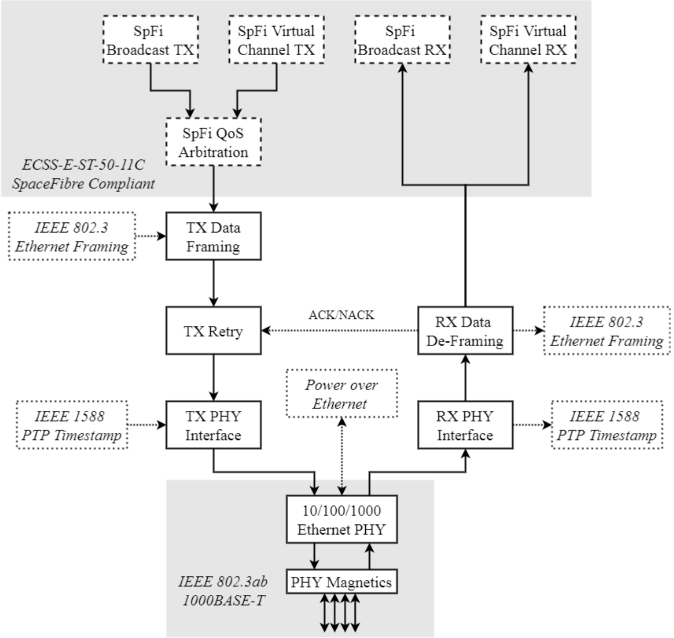


Fig. 1. Overview of the SpacePHYre CODEC. The upper layers of the CODEC implement the SpaceFibre Virtual Channel and Broadcast Interfaces as well as SpaceFibre Quality of Service Mechanisms. The lowest layers of the CODEC interface with IEEE802.3ab 1000BASE-T Gigabit Ethernet PHYs. Several optional features are available and indicated by dashed lines.

power cables, significantly reducing spacecraft mass.

3) ***High-precision time synchronisation:*** The Gigabit Ethernet PHYs used can support a highly-accurate time synchronisation service based on IEEE 1588-2008: Precision Time Protocol. Synchronisation in the region of 100ns is possible due to the PHYs ability to timestamping SOF tokens within the transmit and receive analogue front ends of the PHY. This also removes the need for dedicated PPS cables, reducing spacecraft mass.
4) ***Ethernet Compatibility:*** The use of 1000BASE-T PHYs allows electrical compatible with terrestrial Ethernet networking equipment, allowing easier integration and development of EGSE test equipment and processes.

II. Overview of SpacePHYre CODEC

SpacePHYre implements the SpaceFibre standard (ECSS-E-ST-50-11C [2]) for the Network layer, Virtual Channel layer and Quality of Service provisioning. This provides the same deterministic and real-time properties of SpaceFibre links and ensures compatibility with other devices when used within a mixed SpacePHYre/SpaceFibre/SpaceWire network.

The Framing and Retry layers are modified to support the characteristics of the Gigabit PHYs, but retain the SpaceFibre frame formats, sizes and required frame headers to support the virtual channel and QoS provision.

The Link layer presents full data or control frames to the PHY and also manages the link running state via the PHY MDIO interface or discrete signals. Control logic within the PHY handles link connection, link runtime management, as well as translation of 8-bit data words to/from tokens to be transmitted or received over the link. This removes the need for the link layer of the CODEC to manage low-level control tokens, perform 8b/10b encoding or manage the elastic buffer on the receive data path. This results in a simpler link layer when compared to SpaceFibre.

Figure 1 indicates the arrangement of the CODEC and which parts are from SpaceFibre, which parts are SpacePHYre specific and which parts need to control the PHY as a standard 1000BASE-T (IEEE802.3ab) interface.

A. *Virtual Channel Interface and QoS*

As shown in 1, SpacePHYre fully implements ECSS-E-ST-50-11C for the Virtual Channel interfaces, Broadcast interface and Quality of Service provisioning. This allows compatibility with SpaceFibre networks.

B. *Data Framing Layer*

The same framing control tokens (e.g. VC flow control) and framing primitives are used as a SpaceFibre frame (256 bytes per data frame, 8 bytes per broadcast).

The 8-bit PHY interface does not allow identification between control and data words, therefore the framing of packets is vital to extract the data information from the incoming data stream. For this reason, broadcasts and control tokens cannot be injected into data frames that are being transmitted, instead they must wait until the frame has finished transmitting.

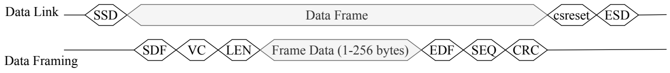


Fig. 2. Virtual Channel Data Framing.

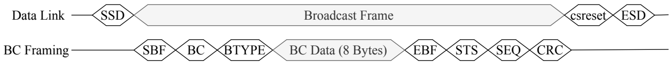


Fig. 3. Broadcast Data Framing.

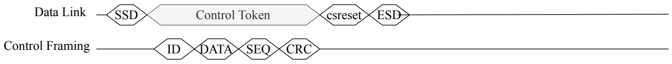


Fig. 4. Control Token Framing.

The frames also require some tokens adding to support transmission over the Gigabit PHY such as SSD (start of stream), ESD (end of stream) and CSRESET (reset of collision sense logic). In the case of an idle link and singular frames then these tokens are added to the header and tail of the frames. If multiple frames are ready for sending then these can be combined, as shown in Figure 5, subject to the maximum stream length characteristics of the particular link.

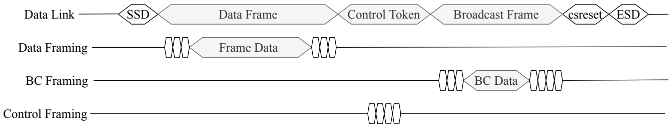


Fig. 5. Example of multiple frames being packed together for continuous transmission, only requiring a single set of SSD and ESD PHY tokens for a number of frames.

C. Retry Functionality

The frame retry mechanism is based on SpaceFibre's use of ACK and NACK tokens and resending of frames should a CRC error be detected by the far end. All frames for transmission are stored in a retry buffer and are either removed from the buffer or resent on recipient of an ACK or NACK token respectively.

D. Gigabit PHY Interface

The CODEC can support standard Gigabit PHY interfaces, such as GMII, RGMII and SGMII. Control of the PHY is either via the PHY register interfaces and the MDIO interface or via discrete signals into dedicated control or initialisation pins of the PHY. Management functions via MDIO are asynchronous to the data interface and are slow and sporadic in nature. The MDIO interface can be managed by a small state machine within the CODEC or also by an embedded processor in implementations that include an embedded processor.

E. Power Delivery

The magnetically coupled data lines allow power to also be provided over the same wiring, using the same methodology as Power-over-Ethernet. The power supply and handshaking will be optimised for the application for spacecraft avionics. The PHYs will report if the far end needs powering, then extra features in the Link Management layer provide power handshaking to ensure the correct amount of power is provided to the remote end. The Link Management layer also interfaces with the power supply at the source end of the link to enable and control the power delivery over the link.

F. Further Optional Features

The use of Gigabit Ethernet PHYs adds some new possibilities that complement the existing features of SpaceFibre.

1) IEEE-1588 PTP Timestamping: Precision Time Protocol is a network time synchronisation protocol that allows very accurate time synchronisation. It achieves this through accurate measurements of link latencies and applying corrections for these latencies. Time synchronisation packets are timestamped as they are transmitted and received by a link. The less jitter between the actual link latency and these timestamps, the greater the precision of the achieved time synchronisation. Some Gigabit PHYs allow very accurate time stamping by detecting the time synchronisation frames within the front-end electronics of the PHY. This can be used to perform very accurate time synchronisation across a SpacePHYre link.

2) Ethernet Frame Compatibility: The use of Gigabit Ethernet PHYs and transformer magnetics gives electrical compatibility with terrestrial 1000BASE-T equipment, but not Ethernet compatibility due to the lack of a MAC frame header. An optional feature of the framing layer allows SpaceFibre frames to be wrapped with IEEE 802.3 Layer 2 Ethernet frame headers and so compatible at the packet level with Ethernet networking products. This removes the need for adaptor bricks for PC to Spacecraft communication during AIT and development activities. The PC would then be responsible for packaging SpacePHYre frames within these Level 2 Ethernet frames such to fulfil the operating requirements of SpacePHYre. In this use, QoS primitives would not be guaranteed to hold due to the non-deterministic operation of Ethernet MACs.

3) Energy Efficient Ethernet: The IEEE 802.3az standard gives PHYs the ability to detect when a defined period of no data transmission has occurred [6]. The PHY can automatically power down the transmit circuitry of the PHY into a standby state once this period has occurred. The receiver circuitry is kept active and so the link is quickly re-established once data is ready for transmission.

III. Prototype Implementation

A prototype SpacePHYre codec is under development. A demonstration PCB hosting two Microchip VSC8541 PHYs (the commercial equivalent part of the VSC8541RT) has been developed for the Ultra96 FPGA board. The CODEC is being developed within the FPGA and tested with this PHY. The VSC8541 can be configured on device power up via the bootstrap pins or via the MDIO interface. The prototype CODEC will showcase demonstrate functionality without the MDIO interface at first, a MDIO controller will be added in a later implementation.

In addition to the VSC8541 prototype, power delivery daughter boards have been developed to demonstrate Power over Ethernet to the IEEE 802.3at standard [6]. Daughter boards for both a power source and a powered sink have been developed based on a COTS fully isolated 802.3at PoE solution. This allows up to 25W to be delivered over the cabling, using up to 57V injected into a pair of the data cables.

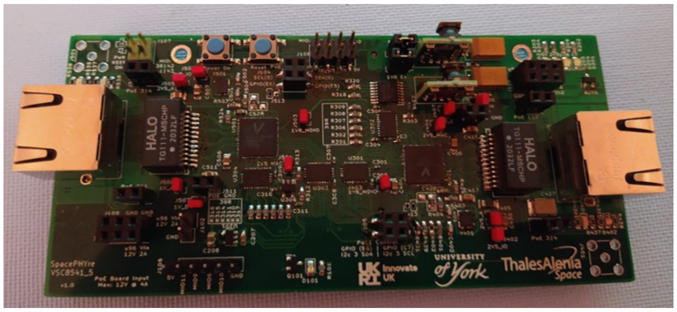


Fig. 6. Prototype SpacePHYre PCB supporting two Microchip VSC8541 PHYs (the commercial equivalent part of the VSC8541RT) and matched magnetics, the board is plugged in to an Ultra96 FPGA board where the CODEC is implemented within the FPGA hardware.

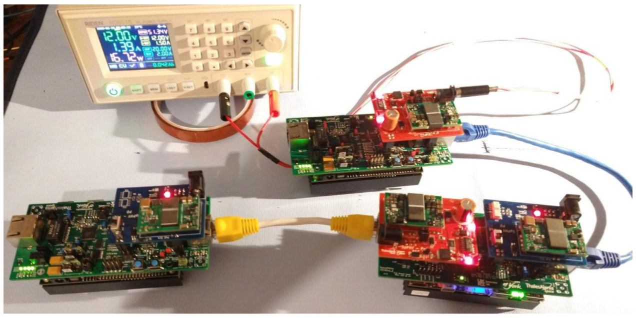


Fig. 7. Power Delivery Test. The prototype board is extended with power source and power sink boards to carry IEEE 802.3at PoE over the magnetics. Up to 25W can be delivered.

IV. Further Development

Several developments for SpacePHYre are currently being undertaken at TAS-UK or planned for the near future:

1) *VHDL IP Core:* The SpacePHYre CODEC is currently being developed as a VHDL IP core for full testing and evaluation with the VSC8541. It is intended to release a prototyping version of this IP to the Space community in due course
2) *Testing with DP83561-SP PHY:* No commercial equivalent PHY exists for the TI PHY and so an evaluation will need to be made with SpacePHYre IP core and the evaluation board for this device.
3) *Develop and define standards for power delivery:* The IEEE standards for Power-over-Ethernet are focussed on fire and electrical safety in office environments and inline with terrestrial wiring standards. It is expected that efficiencies in operation can be made by tailoring for space applications only. Part of this work would also develop a handshaking and negation procedure, such that the power delivery can be managed and controlled as the required by the space application.
4) *Measurement of time synchronisation precision achievable with PTP:* It is expected that time synchronisation to 10ns can be achieved using PTP and the SoF detection features of the PHYs.
5) *Evaluate the performance of SpaceWire cables:* SpaceWire cables and connectors provide the required number of twisted data pairs for transmission between Gigabit Ethernet PHYs, a study will determine their suitability for Gigabit transmission. Their suitability and limitations for power delivery will also be assessed.

V. Conclusions

SpacePHYre interfaces will complement existing SpaceFibre and SpaceWire networking by providing a significantly more robust interface, well suited to operation over long cable runs or with equipment that can generate large transients between the two endpoints. The power delivery opportunity presents large savings in harness mass and new technological development opportunities for powering remote units. It is envisioned that SpacePHYre could provide isolation between the platform and instrument avionic networks or for instruments that may operate on the periphery of the spacecraft, with SpaceFibre being used for the higher-speed interconnects required within electronics units e.g. for backplanes. The use of magnetic isolation enables an even greater degree of modular design and reuse, as better guarantees can be made of the isolation between modular units that AC coupling cannot provide.

The technology is in active development and, thanks to the recent availability of flight-suitable Gigabit PHYs, a technology demonstrator of the full CODEC is expected to be presented soon.

References

[1] STAR-Dundee Limited, *SpaceFibre Interface IP Core Datasheet*, 2022.

[2] ECSS, "SpaceFibre Standard - ECSS-E-ST-50-11C," ECSS, Standard, 2019.

[3] Microchip, *VSC8541RT - Radiation-Tolerant Single Port Gigabit Ethernet Copper PHY with GMII/RGMII/MII/RMII Interfaces*, 2019.

[4] T. Instruments, *DP83561-SP Radiation-Hardness-Assured (RHA), 10/100/1000 Ethernet PHY Transceiver with SEFI Handling Sub-System*, 2021.

[5] TTTech, *TT6802-1-SE - The TTEEnd System Controller Space*, 2022.

[6] IEEE, "IEEE Standard for Ethernet, IEEE 802.3-2018," IEEE, Standard, 2018.

SpaceWire hardware abstraction layer Considerations

Mitsutaka Takada
Center for Embedded Computing Systems,
Graduate School of Informatics(NCES)
Nagoya University
Nagoya, Japan
mtakada@nces.i.nagoya-u.ac.jp

Takayuki Ishida
Insititute of Space and Astronautical Science (ISAS)
Japan Aerospace Exploration Agency (JAXA)
Sagamihara, Japan
ishida.takayuki@jaxa.jp

Hiroaki Takada
Center for Embedded Computing Systems,
Graduate School of Informatics(NCES)
Nagoya University
Nagoya, Japan
hiro@ertl.jp

Keiichi Matsuzaki
Insititute of Space and Astronautical Science (ISAS)
Japan Aerospace Exploration Agency (JAXA)
Sagamihara, Japan
matsuzaki.keiichi@jaxa.jp

Abstract—It is necessary for spacecraft software to be developed faster while ensuring reliability through mass production and reduced development turnaround time. The formulation and implementation of software platforms, as well as the standardization of onboard data communication applications, are techniques for improving software development efficiency. In addition, communication middleware and platforms have been developed, and it has been discussed and described by The Consultative Committee for Space Data Systems (CCSDS SOIS).

On the contrary, hardware designs are frequently improved for greater speed, functionality, and reliability. The hardware design changes and purchase of devices from various manufacturers result in driver updates and software interface changes between the drivers and middleware. In spacecraft development, model-based, simulation, and target-hardware board development are conducted for each process. Developing and verifying using both middleware and actual board drivers are backward processes.

Therefore, this study examined the SpaceWire hardware abstraction layer that can be used with multiple actual target boards and middleware. Furthermore, we developed and verified an abstraction layer driver that can be used with various middleware and host OSs to validate.

Index Terms—SpaceWire, Software platform, Hardware abstraction layer, Device driver

I. Introduction

Spacecraft onboard networks are becoming increasingly important due to the increasing size of CubeSats and the growing scale of mission data. Onboard network applications use CCSDS-compliant protocols for inter-application communication, so onboard software, including communication applications, communication middleware, device drivers, and onboard OS, work together to form communication stacks (Fig. 1). Since onboard networks are required to be faster, more functional, and reliable, design changes are made to speed up hardware or convert parts of the communication stack that are implemented in software into hardware. As a result, the “device driver” software, which controls communication devices, is forced to adjust and change the interface between the onboard OS and communication middleware according to the hardware design changes.

Since SpaceWire devices are easy to implement using FPGAs, they are now available as onboard devices and as discrete expansion boards for consumer products like the Raspberry Pi. The environment is prepared to develop spacecraft and flight products using Hardware In the Loop Simulation and Proof of Concept on the ground. While hardware componentization will continue to accelerate in the future, satellite software standardization has already been accomplished for protocol and service-based standards, like CCSDS SOIS [1]. However, the onboard software platform for each satellite development project varies.

For instance, using a discrete SpaceWire product, even when the OS and communication middleware development process and development costs involve several software platform environments. Device driver functionality and performance should not vary when using the same SpaceWire device; hence the device driver’s source code must remain unchanged.

The following studies were conducted to lower the barrier to introducing SpaceWire.

- Classification of the currently available SpaceWire device drivers and driver interfaces from communication middleware.
- We studied the SpaceWire hardware abstraction layer, which is the standard driver interface between the communication middleware and the device, and developed a driver interface for SpaceWire devices.

- We developed a new SpaceWire interface and device driver for the SpaceWire / RMAP library based on the SpaceWire device running in the Raspberry Pi + Linux environment and developed a new device driver for the SpaceWire RMAP library.
- We validated the study's communication using the device driver.

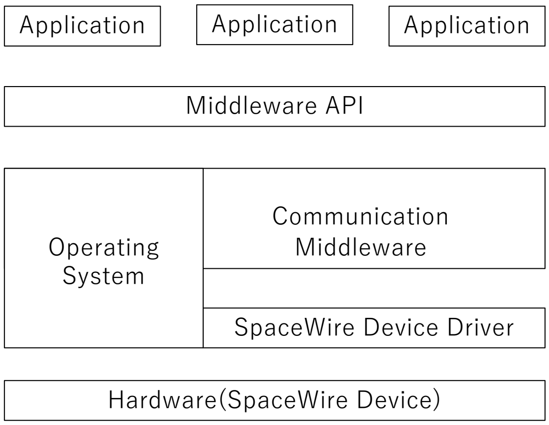


Fig. 1. Example of SpaceWire communication stacks.

II. SpaceWire driver interfaces classification

The following are the main hardware interfaces we have used as software-controlled SpaceWire devices.

1) Onboard CPU bus connection.
2) Connection via the onboard external bus.
3) Other connection methods.

Table I shows the results of classifying these device types from the following perspectives.

- SpaceWire protocols supported by the hardware.
- Software access to the device.
- How headers and payloads are configured when constructing SpaceWire packets.
- Interrupt notification from the device.

Except for No d, we found that devices are accessed via memory-mapped I/O when directly connected to the onboard CPU bus. In contrast, discrete expansion devices can be accessed via general-purpose I/O like SPI.

III. Interface design of communication middlewares

We have developed and implemented communication middleware with a SpaceWire devices interface. The features of each communication middleware and the methods of accessing the devices are listed below.

A. SpaceWireOS

SpaceWireOS is a platform consisting of RTOS, communication middleware, and a communication management table based on SpaceWire-D, which was developed in a joint research project between Nagoya University and JAXA/ISAS [3].

In order to shorten the time required to implement SpaceWireOS on a CPU, the communication middleware and SpaceWire device driver were developed in parallel, and the communication middleware was developed by creating stubs in the driver section in advance that simulated the target hardware.

Therefore, the approach was to define the Application Programming Interface between the middleware and the device driver in advance and then create glue code for the target hardware in the API processing section at the time of integration. Specifically, the middleware part uses stubs to check middleware functions. The driver part checks primitive operations and communication on the board and then develops and verifies the functionality of the coupling checks.

Since the board with SpaceWireOS was equipped with essential hardware functions for both receiver and transceiver based on RMAP, we defined primitive functional configurations and APIs for the communication middleware and device driver.

B. Software Bus Network

Software Bus Network (SBN) is one of the core Flight System applications being developed by NASA and has the following features [5].

- Connects to other SBN applications via P2P.
- Receives messages from other bus applications.
- Supports network architectures such as TCP, UDP, and Serial.
- Announcing and heartbeat functions for network state awareness.
- Configuration table for outgoing messages and filtering.

SBN has the source code for the SpaceWire Interface. However, due to using the prototype code and socket IF, the original code of SBN cannot use SpaceWire communication. We have ported and verified the operation to the SBN using SpaceWire devices that connect to consumer electronics devices. They also studied configuration files for SpaceWire that can be used in the SBN [6].

TABLE II summarized the interface name of the SBN when we tried to implement it, the UDP interfaces used, and the corresponding interfaces of the SpaceWire we implemented.

C. SpaceWire / RMAP Library

It was developed as a common software library that handles functions related to SpaceWire / RMAP [7]. The library is written in the C++ language. It is implemented with few dependencies on external libraries, making it usable in general-purpose OS (Linux, macOS) and RTOS environments that can handle POSIX. TABLE. III shows the overall structure of the library. The SpaceWireIF class is positioned as an abstract wrapper class for the

TABLE I
Lists of SpaceWire device types

No	Device type	HW support protocol	Addressing	Setting header	Setting payload	Notification of interrupt
a	Include OBS	RMAP	CPU register, Memory-mapped I/O	Memory-mapped I/O	Memory-mapped I/O	RX,TX complete
b	Include OBS	PTP, RMAP, SpaceWire-D, SpaceWire-R	CPU register, Memory-mapped I/O	Memory-mapped I/O	Memory-mapped I/O	RX,TX complete
c	Ediscrete expansion board	PTP	SPI	SPI	SPI	RX complete
d	Other	PTP	Socket	Socket Application	Socket Application	-

TABLE II
Support SpaceWire IF API for SBN

IF name	UDP	SpaeWire
InitModule	SBN_UDP_Init	SBN_SPW_Init
LoadNet	SBN_UDP_LoadNet	SBN_SPW_LoadNet
LoadPeer	SBN_UDP_LoadPeer	N/A
InitNet	SBN_UDP_LinitNet	SBN_SPW_Init
InitPeer	SBN_UDP_InitPeer	N/A
PollPeer	SBN_UDP_PollPeer	N/A
Send	SBN_UDP_Send	SBN_SPW_Send
RecvFromPeer	N/A	N/A
RecvFromNet	SBN_UDP_Recv	SBN_SPW_Recv
UnloadNet	SBN_UDP_UnloadNet	N/A
UnloadPeer	SBN_UDP_UnloadPeer	N/A

SpaceWireIF hardware and OS-loaded driver. The following is a partial list of the SpaceWireIF class methods.

TABLE III
SpaceWire device interface of RMAP Library

SpaceWireIF class method	Discription
open()	Open IF
close()	Close IF
send()	Send Packet
receive()	Receive Packet
emitTimecode()	Emit TimeCode
setTxLinkRate()	Set Tx link rate
setTimeoutDuration	Set timeout packet
getTimeCode	Get TimeCode

IV. Consideration of SpaceWire Device Driver and Hardware Abstraction Layer

The following requirements for the SpaceWire Device Driver and Hardware Abstraction Layer (HAL) have been summarized. The SpaceWire Device Driver and HAL are the Data Link Layer of the SpaceWire specification [8].

- The IF specification should satisfy the requirements of the upper Network Layer (NL), the lower Encoding Layer, and the Management Interface Base.
- The upper NL has different implementation languages depending on the use case and purpose, but the HAL has an interface with the upper layer capable of handling SpaceWire packets, broadcast codes, and time codes.

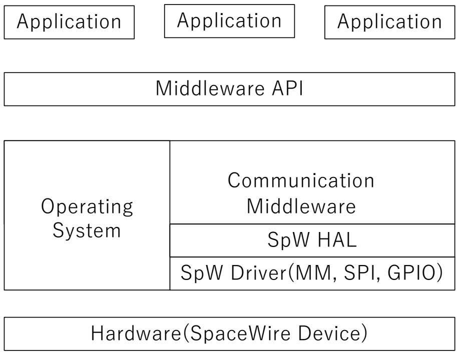


Fig. 2. SpaceWire communication stacks with SpaceWire HAL

- The HAL identifies lower-layer devices by SpaceWire Port.
- If there are multiple SpaceWirePorts, the driver of the SpaceWire Port corresponding to the port specified by the upper layer is configured appropriately.
- The HAL does not perform data transfer between SpaceWire Ports. If routing is required, it is handled at the upper layers.
- The device driver source code should be reusable regardless of OS or middleware.

The following implementation requirements and constraints were added to simplify software processing.

- No memory area is allocated for storing data. If the system uses the memory area prepared by the upper layer or if the upper layer application can access the memory area allocated by the lower layer, the address of the memory area shall be notified to the upper layer.
- If the lower layer can obtain the link information, the HAL does not need to maintain the link information.
- If the lower layer can process the transmission priority, the HAL and the device driver do not need to process the transmission priority.

We decided to further divide the SpaceWire interface into the following two layers based on the above requirements.

SpaceWire interface is further divided into the following two layers.

- SpaceWire HAL.
- SpaceWire device driver layer.

The HAL connects the SpaceWire Port to the underlying device driver layer (DDL). The DDL handles initialization and data transmission from the SpaceWire device and performs configuration processing dependent on the OS and middleware. The OS-independent processing code is implemented by describing the I/O process that accesses the device to the extent that it is OS and middleware-independent.

V. Implementation and conclusion

We developed a SpaceWire HAL and device driver based on the SpaceWire HAL study using two distinct OS and middleware (Linux + RMAP Library and RTOS + SpaceWireOS) on the same hardware. This hardware controls SpaceWire via SPI. With the HAL interface implementation code, We verified that the device driver part of the lower layer could absorb the differences between the OS and middleware without any changes in the SPI control code. We were able to confirm SpaceWire's communication with each other.

TABLE IV
SpaceWire HAL implementation

HAL IF	RMAP Library	SpaceWireOS
Open	open	target_initialize_spwd
Close	close	–
Send	send	rmap_send_cpacket rmap_send_rpacket
Receive	receive	rmap_receive_packet
Emit TimeCode	emitTimecode	–
Get TimeCode	getTimeCode	spw_get_timecode

Since multiple middlewares have different purposes, unifying the standard SpaceWire HAL interface API common to all middlewares was impossible. Still, we could summarize the interfaces and their parameters for each function. We could also summarize the interfaces for operating SpaceWire devices without depending on actual IO for the lower layer device drivers. It is necessary to consider cases where networks other than SpaceWire, such as SBN, are also supported. In the future, we would like to increase the number of device implementation examples and expand verification tools like device testing.

References

[1] CCSDS: CCSDS 850.0-G-2, Spacecraft Onboard Interface Services, Washington, USA, CCSDS 2013

[2] University of Dundee:SpaceWire-D Standard, Draft D, Issue 0.15, 2014

[3] M. Takada, H. Takada, Y. Chen, T. Yuasa, T. Takahashi and M. Nomachi, Development of software platform supporting a protocol for guaranteeing the real-time property of SpaceWire, International Space Wire Conference, June 2013

[4] S. Parkes, A. Ferrer, S. Mills and A. Mason, SpaceWire-D: Deterministic data delivery with SpaceWire, International SpaceWire Conference, June 2010

[5] Knight, Christopher D. Core Flight System Software Bus Networking Application Design as Built, Workshop on Spacecraft Flight Software (FSW-16), No. ARC-E-DAA-TN37567, 2019

[6] Takada, Mitsutaka, et al. Porting cFE to spacecraft onboard with SpaceWire Engine and RTOS based on uITRON specification, FSW2017, 2017

[7] T. Yuasa:SpaceWire RMAP Library v2 User Guide, 2012

[8] Secretariat, SpaceWire-Links, nodes, routers and networks, ECSSE-ST-50-12C, Noordwijk, 2008

Multi-bus protocol Controller Based on SpaceWire

Meng Zou
Beijing Microelectronics Technology Institute
Beijing, China
zmseu2013@163.com

Xingyou Wang
Beijing Microelectronics Technology Institute
Beijing, China

Yuehua Niu
Beijing Aerospace Vehicle General Design Department
Beijing, China

Haidong Fei
Beijing Microelectronics Technology Institute
Beijing, China

Wei Zhuang
Beijing Microelectronics Technology Institute
Beijing, China

Guowei Hou
Beijing Microelectronics Technology Institute
Beijing, China

***Abstract*—In order to meet the needs of data transmission among multiple devices in satellite onboard equipment, the multi-bus protocol controller based on SpaceWire provides 4 SpaceWire ports, 1 host control interface, 6 general-purpose UART interfaces, 16 SPI master interfaces, 4 SPI slave interfaces and 2 I2C interfaces to implement data interaction between SpaceWire interface and host control interface, SPI interface, UART interface and I2C interface.**

The multi-bus protocol controller is compatible with the latest SpaceWire standard ECSS-E-ST-50-12C Rev.1 and also supports the ECSS-E-ST-50-51C, 52C and 53C protocols. This controller can monitor SpaceWire link data flow and is capable of error detection and retransmission of SpaceWire packets to ensure the reliability of data transmission. The multi-bus protocol controller supports the transmitting and receiving of broadcast codes and also provides the function of slot planning based on timecode to enable data deterministic transmission.

The multi-bus protocol controller can operate at data-rates between 2Mbps and 400Mbps per SpaceWire port and it has 64 time slots with 8 breakpoints in each time slot for fine deterministic and effective data processing capabilities. The host control interface has a large FIFO capacity of 16K bytes for both transmitter and receiver and the interface bus width can be configured as 8/16/32 bits. The UART interface can be connected to RS232, RS422, RS485, LVDS, M-LVDS transceiver circuits to form a powerful multi-bus protocol network.

Keywords—SpaceWire, deterministic transmission, multi-bus protocol, high reliability

I. Introduction

With the growing process capability of spacecraft data handling system, the data conversion between various protocols has become more complex, especially in the case of high real-time requirements, so the control function of protocol conversion has become critical. At present, the existing SpaceWire controllers can only connect to FPGA or CPU, and most applications use FPGA to implement the control logic between SpaceWire and other interface protocols. The multi-bus protocol controller as an aerospace-grade ASIC circuit, with small size, low power consumption, anti-irradiation characteristics, can achieve protocol conversion function based on the SpaceWire.

The multi-bus protocol controller with 4 SpW(SpaceWire) ports which are compatible with ECSS-E-ST-50-12C Rev.1 [1] and support the send rate of 2~400Mbps, can accomplish point-to-point high-speed data transmission between multiple devices. Each SpW port supports packet length truncation function, packet and character statistics function, path address and logical address routing, group adaptive routing. One SpW monitoring port can be configured to monitor and collect data flow from any one of the 4 SpW ports.

The multi-bus protocol controller has one HOCI(Host Control Interface), which can be easily connected to the user control processing unit and can be convenient for users to control the operation of data and register access. The multi-bus protocol controller has several local interfaces including UART, SPI master, SPI slave and I2C, which support CLTP(Controller Local Transmission Protocol) and the local interfaces can communicate with SpW ports and HOCI. Thus, implementing the conversion function of a variety of protocols, which is simple and convenient for equipment to use.

The multi-bus protocol controller has the automatic retransmission function based on the RDDP(Reliable Data Delivery Protocol) [2] to ensure the reliable transmission of packets, also supports timecode sending and forwarding functions and distributed interrupt/acknowledgment functions. Additionally, deterministic transmission based on timeslots can be acquired from this circuit.

Fig. 1. Physical diagram of the multi-bus protocol controller

The physical diagram of the multi-bus protocol controller is shown in Fig. 1, the design of the device adopts radiation-hardened process and logic, has the feature as anti-single event upset and anti-latch-up capability, ESD protection above 2kV Human Body Model, package form is CBGA256, package size is 21mm x 21mm x 3mm, power consumption is less than 1.5W, operating temperature is from -55 °C to +125 °C, complied with aerospace-grade conditions.

II. System Specification

A. Architecture overview

As shown in Fig. 2, the multi-bus protocol controller is composed of SpW routing module, protocol conversion module, HOCI, local interface, register set, broadcast interface and GPIO(General Purpose Input Output).

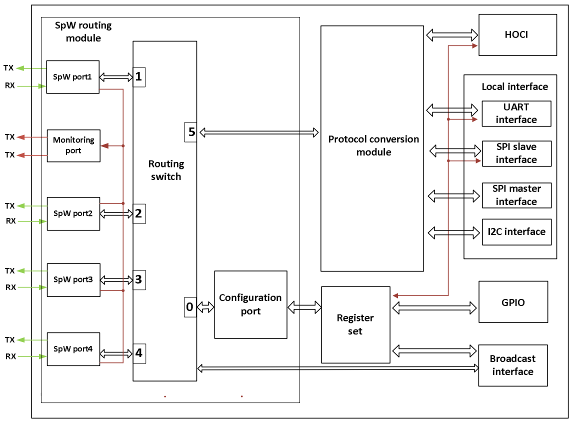


Fig. 2. Multi-bus protocol controller Architecture

Among them, the SpW routing module consists of four SpW ports, monitoring port, routing switch and configuration port. The local interface includes UART interface, SPI master interface, SPI slave interface, and I2C interface. The summary of the characteristics of each module is shown in TABLE I.

TABLE I. Characteristics of Modules

Module Name	characteristics		
	Port number	*Maximal speed per port*	*Cache capacity*
SpW port	4	400Mbps	---
UART	6	1Mbps	2K Bytes
SPI slave	4	10Mbps	1K Bytes
SPI master	16	10Mbps	1K Bytes
I2C	2	400Kbps	1K Bytes
HOCI	1	50MHz × 32bits	16K Bytes

As can be seen in TABLE I, the local interface has six UART ports, sixteen SPI master ports, four SPI slave ports and two I2C ports. There is a transmit FIFO and a receive FIFO in each port, which are the same in capacity and can store each complete packet data for efficient transmission, the transmit FIFO capacity or the receive FIFO capacity is named as cache capacity.

The SpW routing module receives the SpaceWire signal from the external device, by the routing switch, routing to the protocol conversion module or configuration port. Through the protocol conversion module, the data can be passed to the HOCI or local interface (UART, SPI master, SPI slave, I2C), through the configuration port, using the RMAP(Remote Memory Access Protocol) packet format [3], the register set can be accessed. Similarly, HOCI data can be routed to the SpW port or the local interface through the protocol conversion module. Of course, the data of the local interface can also pass through the protocol conversion module to reach HOCI or SpW ports.

The register set inside the multi-bus protocol controller can not only configure the rate, operation mode, transmission direction, protocol identification and other parameters of each module of the controller, but also query the capacity, statistical quantity, error status etc., so as to realize the normal communication function, such as the SpW monitoring port, by the register, can monitor the traffic of any SpW port and the 32-channel GPIO with multiplexing function can be used by the user according to different application scenarios through the configuration register.

The multi-bus protocol controller has several flexible methods for register set access, which is summarized below.

- All SpW ports via RMAP packet.
- All UART interfaces.
- All SPI slave interfaces.
- HOCI

The broadcast interface of the multi-bus protocol controller supports the transmission and reception operations of timecode and distributed interrupt/ acknowledgment, and the transmission of timecode has two ways: configuration register or external trigger, and the timecode is forwarded through the SpW port to realize the information synchronization function in the SpaceWire network system.

B. CLTP

The ECSS-E-ST-50-51C standard stipulates that protocol identification [4] from 240 to 254 (0xF0 to 0xFE) are used for user-defined protocols, and based on this range of protocol identification numbers, the CLTP of the controller implements the data interaction functions of the local interface(UART, SPI master, SPI slave, I2C) and HOCI or SpW port.

The CLTP packet format is similar to the STUP packet format [5], as shown in Fig. 3.

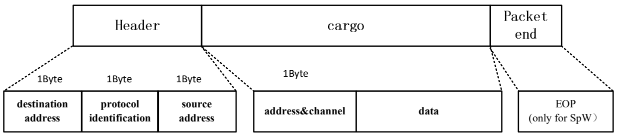


Fig. 3. CLTP packet format

For each module of UART, SPI master, SPI slave, I2C, SpW port and HOCI, the multi-bus protocol controller has the corresponding destination address register, source address register, protocol identification register, which can be modified by register access. The destination address and source address can be both logical and path addresses, which are used for the guidance of data routing. When the local interface such as UART's protocol identification is equal to HOCI, entire packet of data from UART are routed to the HOCI through the protocol conversion module, if the protocol identification of the UART and the SpW port are equal, the protocol conversion module routes the entire packet of data to the SpW port, meanwhile, SPI master, SPI slave, I2C is in the same way, to determine whether the data flows to HOCI or SpW port.

The second byte of the packet from the SpW port is the protocol identification, if the protocol identification is equal to the value in protocol identification register of SpW port, the protocol conversion module routes the entire packet of data to the local interface, and the specific route to which port requires the fourth byte address of the packet to be parsed, and the byte interpretation of address & channel is shown in TABLE II. Packets from HOCI should also compare the protocol identification carried with them with the protocol identification of HOCI, and if the values are identical, data is routed to the local interface, the specific route to which port also needs to be determined by the byte of address & channel in the packet.

TABLE II. BYTE INTERPRETATION OF ADDRESS & CHANNEL

Bit[7:6]	Bit[5:0]
0	0 : uart0 selected 1 : uart1 selected … 4 : uart4 selected 5 : uart5 selected other values : reserved
1	0 : spim0 selected 1 : spim1 selected … 14 : spim14 selected 15 : spim15 selected other values : reserved
2	0 : spis0 selected 1 : spis1 selected 2 : spis2 selected 3: spis3 selected other values : reserved
3	0 : i2c0 selected 1 : i2c1 selected other values : reserved

The byte of address & channel can be divided into two parts, the most significant two bits determine which module of the local interface is selected, the least significant six bits determine the specific port of the module selected by the most significant two bits, and the values of Bit [7:6] and Bit [5:0] in TABLE II are in decimal format.

C. Transmission between SpW port and HOCI

SpW ports support packet length truncation which can be implemented by register configuration. The maximal length of truncation can reach 16K Byte. Packet and character statistics function for each SpW port is also supported. Both input port and output port can be counted, and the statistical results can be viewed by accessing the register, which is convenient for users to query the number of packets and data bytes. SpW routing module allows path address and logical address to pass through the routing switch for routing function between different SpW ports, meanwhile group adaptive routing is also supported in SpW routing module.

There are two ways to transfer data from the SpW port to HOCI: one is that the protocol identification between SpW port in register and the value carried in the data from the SpW port are not equal, then the data will be routed to HOCI, and the other is by setting the transparent bit enable in the register, at this time, the data of the SpW port flows directly to the HOCI, and the protocol identification is irrelevant, and the data will not enter the local interface.

HOCI supports configurable 8/16/32 bits bus width, big-endian or little-endian mode, which is flexible for use to accomplish different processors. HOCI has transmit FIFO and receive FIFO, as described in table1, the cache capacity of HOCI is 16K byte. In addition, HOCI supports efficient data transmission and auto retransmission function.

HOCI sends three types of packets, namely HOCI to SpW port normal packets, HOCI to SpW port RDDP packets, HOCI to the local interface packets, these three types are distinguished by the control word sent by HOCI, Packets that are sent from HOCI to the local interface in addition to the control word also need CLTP format content to fill in the sending packet.

In addition, if the protocol identification carried in the packet from HOCI is not equal to the protocol identification of HOCI, which stored in register, the packet is considered as a normal packet and is routed to the SpW port through the protocol conversion module.

D. Deterministic and reliable implementation

The multi-bus protocol controller has the function of SpaceWire-D [6], which is implemented on protocol conversion module, when data are sent from HOCI to SpW port. This way of sending data is based on the broadcast timecode in the SpaceWire network system.

The timecode value broadcast by the system is 0 ~ 63, so the entire system corresponds to a total of 64 time slots. Each time slot includes eight breakpoints, and each breakpoint has a register correspondingly, each breakpoint register can independently set the start and stop of data transmission, produce interrupt status and other information, so that the transmission of data in a time slot can be configured to accomplish high real-time and determinism.

Each time slot has eight breakpoint registers, interrupt status registers and maximum length register of transmitting data, to achieve deterministic transmission in time slots, multiple packets of data can be sent in a time slot, the time interval of all time slots can be set through the global timer register. The trigger condition of deterministic transmission is that the current received timecode is valid and the time slot corresponding to the received timecode has been completely set. If the current time slot of the data being sent is updated by the next time slot, the unsent data is cleared and an EEP is added to the tail of the data that has already been sent.

In addition, deterministic transmission is only for HOCI to SpW data transfers, and there is no time deterministic feature for data transfers between HOCI and local interfaces.

The multi-bus protocol controller supports automatic retransmission function which is based on the protocol called RDDP, the principle of this function is that retransmission packet format is embedded in the content area of the SpW packet format, through the CRC check, packet sequence number, active reply and timeout mechanism and other measures to detect and recover lost packets, out-of-order packets and bit-err packets.

If the CRC and packet sequence number are correct, the receiver returns a correct state to the sender. Otherwise, if the check results of CRC and packet sequence number are incorrect, the wrong packet is discarded, and no reply is sent. When the sender does not receive the reply packet within the timeout interval set by the register, the packet is retransmitted.

III. Testing Platform

In order to fully verify the function and performance of the multi-bus protocol controller, the construction of the testing platform is particularly important. The verification system is shown in Fig. 4 and is composed of a multi-bus protocol controller evaluation board, a SpaceWire Link Analyser Mk3, a SpaceWire Conformance Tester Mk2, a 400Mbps STAR-System board, an AT7911E board, an AT7910E board, 10m SpaceWire cables and two host computers.

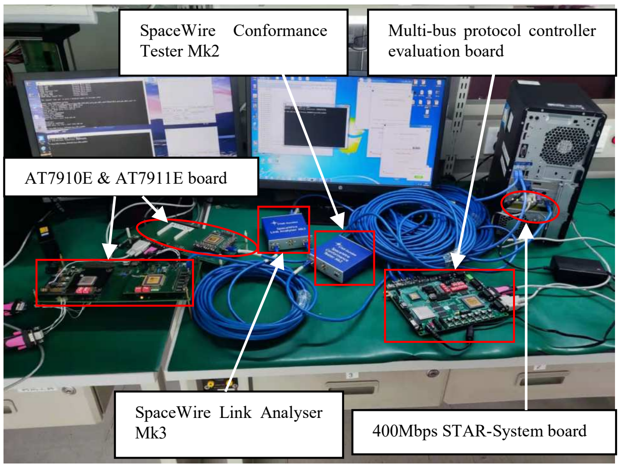

SpaceWire Conformance Tester Mk2
Multi-bus protocol controller evaluation board
AT7910E & AT7911E board
SpaceWire Link Analyser Mk3
400Mbps STAR-System board

Fig. 4. Verification system

The multi-bus protocol controller evaluation board consists of multi-bus protocol controller, FPGA, SPARC processor, SRAM, RS232, RS422, RS485, LVDS, M-LVDS, and other peripherals. The SPARC processor is connected to the HOCI of the multi-bus protocol controller to transmit and receive data; The FPGA is connected to the UART, SPI master, SPI slave, and I2C of the multi-bus protocol controller to realize the verification of the local interface, and the UART of the multi-bus protocol controller connects to RS232, RS422, RS485, LVDS, M-LVDS and other devices, which can realize the exchange of data of multiple protocol interfaces.

The SpaceWire Conformance Tester Mk2 can be used to verify the compliance of the multi-bus protocol controller against the SpaceWire ECSS standard. The SpaceWire Link Analyser Mk3 can display captured data at the signal, character or packet levels and monitor the state of the links with a live statistics display. The 400Mbps STAR-System board can transmit and receive packets at the rate of 400Mbps and can debug SpaceWire device, which is suitable for multi-bus protocol controller to verify complex function. The AT7911E and AT7910E board are 200Mbps products in the same category, which can be used to communicate with the multi-bus protocol controller and verify compatibility. Two host computers are used for debugging between multi-bus protocol controller and other devices. In addition, all tests on the platform are carried out with SpaceWire cables of 10 meters.

IV. Conclusion

The multi-bus protocol controller adopts CLTP, which can realize the conversion function of multiple protocols based on SpaceWire bus interface among aerospace devices, to form a powerful multi-protocol network structure through SpW port and external device interconnection, deterministic transmission and automatic retransmission function make the transmitted data have high efficiency and high reliability. Multi-bus protocol controller has the characteristics of small size, low power consumption, anti-irradiation, etc., and can be widely used in various types of aerospace projects.

References

[1] European Cooperation for Space Standardization, “Space engineering- SpaceWire -Links, nodes, routers and networks,” ECSS-E-ST-50-12C Rev.1 DIR3, November 2015.

[2] M. Gardner et al., Joint Architecture Standard (JAS). Reliable Data Delivery Protocol (RDDP) Specification. Sandia National Laboratories, Report SAND2011-3500, May 2011.

[3] European Cooperation for Space Standardization, “SpaceWire - Remote Memory Access Protocol. ECSS-E-ST-50-52C,” Feb. 2010.

[4] European Cooperation for Space Standardization, “Space engineering -SpaceWire protocol identification,” ECSS-ST-E-50-51C, 5 February 2010.

[5] EADS Astrium GmbH, “STUP SpaceWire protocol specification,” SMCS-ASTD-PS-001, July 2009.

[6] S. Parkers, D. Gibson, A. Ferrer. “SpaceWire-D: Deterministic Data Delivery over SpaceWire,” Data Systems in Aerospace, Warsaw, Poland, June 2014.

Components (Long)

Space-grade 1-10 Gbps parallel optical transceivers and fibre optic connectors for SpaceFibre datalinks

Ronald T Logan Jr.
Ruggedized Photonics
Glenair Inc.
Glendale, California USA
rlogan@glenair.com

Davinder Basuita
EU Business Development
Glenair UK
Mansfield, UK
dbasuita@glenair.com

Abstract — We report on development and qualification testing of space-grade high-speed parallel fibreoptic transceivers and mechanical transfer (MT®) optical connectors to support SpaceFibre fibre optic data networks on-board spacecraft. The transceivers employ hermetically sealed opto-electronic hybrid circuits and low-loss optical coupling that provide optical link performance conforming to the requirements of the SpaceFibre physical layer standard at data rates up to 14 Gbps. The transceivers have undergone extensive environmental and radiation testing including shock and vibration, as well as proton, heavy ion and gamma exposure tests. Ruggedized MT parallel optical connectors were also developed in rectangular and circular formats suitable for spacecraft applications.

Keywords—photonic transceiver, fibreoptic connector, SpaceFibre

I. Introduction

The SpaceFibre standard for the fibreoptic physical layer provides for multi-lane “parallel optical” transceivers in addition to single-lane devices, up to rates of 6.25 Gbps at the time of this writing [1], and it is anticipated that higher data rates may be required in future up to 25 Gbps. We developed rugged space-grade 14 Gbps and 25 Gbps parallel optical transceivers utilizing a hermetically-sealed hybrid optoelectronic microcircuit assembly and active optical alignment process. This construction provides for enhanced optical output power per lane of up to +2 dBm at 850nm and increased sensitivity of -12 dBm typical at 10-14 Gbps, leading to optical link budgets of up to 14 dB over the range of -40C to +85C, far exceeding the capabilities of commercial-off-the-shelf (COTS) transceiver units designed for terrestrial datacom applications over this temperature range.

Many benefits in size and mass can be obtained by moving from individual optical contacts, connectors and cabling to multiple-fibre “ribbon cable”. Multi-fibre Mechanical Transfer (MT) connectors were developed in rugged formats in both D38999-style and micro-D connectors. These connectors utilize the standard 12-fibre “MT” contacts, and are available in multi-mode and single-mode physical contact (PC) versions as well as angle-polished-contact (APC) and expanded beam types.

Taken together, these space-grade parallel optics transceivers and parallel fibre-optic connectors provide means to deploy the SpaceFibre physical layer on spacecraft efficiently and without the need for costly component development and qualification testing programs. This brings full realization of the substantial mass reduction possible when replacing heavy SpaceFibre copper cables with extremely light-weight all-dielectric optical fibre cables.

As discussed in a previous paper [2], the use of optical fibre greatly extends the distance possible for transmission of 10 Gbps signals, compared to SpaceFibre copper cables, due to elimination of the coax cable attenuation losses and the regeneration of the electrical signal by the optical transceivers. The intrinsically light weight of the optical fibres, and the lack of metallic shielding typically required for EMI/RFI reduction, as well as the elimination of many grounding and ground-loop issues, bring substantial weight-reduction and other benefits to spacecraft datalinks.

We present the details of the construction, radiation exposure tests and environmental testing for these transceiver modules and optical connectors. The results show that these high-density transceivers and fibre-optic connectors meet or exceed many of the requirements of spacecraft applications and can be employed to enable optical SpaceFibre links onboard spacecraft.

II. Parallel Optical Transceiver Design

A. Design overview

The 4-channel parallel optics transceiver developed is a ruggedized, harsh environment, PCB-mounted photonic transceiver unit providing from 10-14 Gbps per channel, or up to 40-56 Gbps functionality over the full range of -40C to +85C in high shock and vibration environments. It is designed to survive extreme environmental conditions, including radiation, for military, aerospace and industrial applications. A 25-28 Gbps version of the transceiver is also available, as are 12-channel transmitter and a 12-channel receiver module at 10-14 Gbps.

The modules employ a unique hermetically-sealed design for the optoelectronic hybrid components, permitting extended operation or storage in high humidity and vacuum environments. A proprietary active optical alignment technique provides higher output power and improved detector sensitivity than passively-aligned commercial products, resulting in enhanced link margin. The mechanical design is compact and suited to the harsh temperature and vibration environments found in military, aerospace, railway, and industrial applications.

The transceiver is held securely with captive screws using threaded inserts soldered into the host PCB. The optical interface is a 12-fibre MTP® connector socket for ease of use and compatibility with existing network infrastructure.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

The transceiver units are printed-circuit-board (PCB) mounted units with both an optical connector and electrical connector (Figure 1). They are mounted securely to the user circuit board with four screws, so that mechanical stresses are not imposed on the high-speed low-profile electrical connector. The optical connection to the unit is an MTP-style 12-fibre connector. Conduction cooling is possible either to the circuit board or to an adjacent heatsink above the unit insuring adequate heatsinking in vacuum. A finned convection-cooling heatsink is also available for operations with forced-air cooling in air. Alternative heatsink geometries can be easily accommodated.

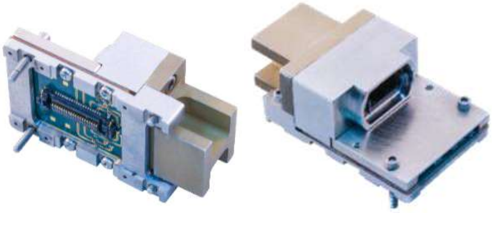


Fig. 1. Parallel optical transceivers with one style of conduction cooling heatsink attached. Bottom view (left) showing electrical high-speed board-to-board connector and top view (right) showing MTP connector receptacle.

Two grades of parts are available for space applications: Radiation-Lot Acceptance Tested (RLAT) and non-RLAT parts. The RLAT parts utilize semiconductor chips from radiation-tested lots, while the non-RLAT parts utilize chips of the same manufacturer, part number and IC fabrication processes, but that have not been lot-tested to verify radiation tolerance. Spacecraft parts are also available with either internal default control of the driver and limiting amplifier ICs, or with external control via I2C and an external user-supplied microcontroller. A version with internal microcontroller is available for non-radiation-applications such as aircraft or ground support equipment.

A functional block diagram of the standard, non-space-grade, 4-channel transceiver with on-board microprocessor is shown in Figure 2. Firmware that is resident in the onboard microprocessor provides for advanced tuning, equalization, and temperature compensation to optimize signal integrity

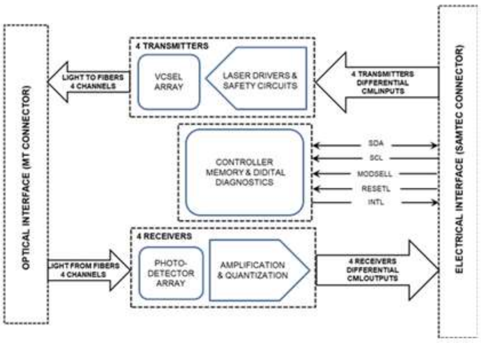

4 TRANSMITTERS
LIGHT TO FIBERS 4 CHANNELS
VCSEL ARRAY
LASER DRIVERS & SAFETY CIRCUITS
4 TRANSMITTERS DIFFERENTIAL CMLINPUTS
CONTROLLER MEMORY & DIDITAL DIAGNOSTICS
SDA
SCL
MODSELL
RESETL
INTL
4 RECEIVERS
LIGHT FROM FIBERS 4 CHANNELS
PHOTO-DETECTOR ARRAY
AMPLIFICATION & QUANTIZATION
4 RECEIVERS DIFFERENTIAL CMLOUTPUTS
OPTICAL INTERFACE (MT CONNECTOR)
ELECTRICAL INTERFACE (SAMTEC CONNECTOR)

Fig. 2. Parallel optical transceiver functional block diagram with internal microcontroller.

B. *Hermetic opto-electronic hybrid*

The transceivers incorporate a hermetically-sealed optoelectronic hybrid circuit that includes the VCSEL array, photodiode array, quad driver IC and quad TIA/Limiting amplifier IC.

The hybrid also incorporates an integrated heating element and thermistor used to stabilize the laser chip array temperature for operation below temperatures of 0 C down to -40C. Active control of the heater permits high-performance operation over the temperature range of -40C to +85C, to support data rates up to 14 Gbps per lane. 25 Gbps-per-lane parts have also been developed and testing is ongoing. The physical partitioning of the functions of the transceiver are depicted in Figure 3.

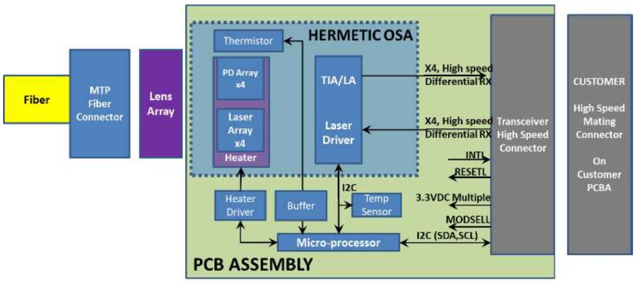


Fig. 3. Parallel optical transceiver physical partitioning block diagram with internal microcontroller. For space applications, the microprocessor is deleted in favor of internal defaults of the laser driver and TIA/limiting amplifier, or external I2C interface to a user's radiation-tolerant microcontroller or FPGA.

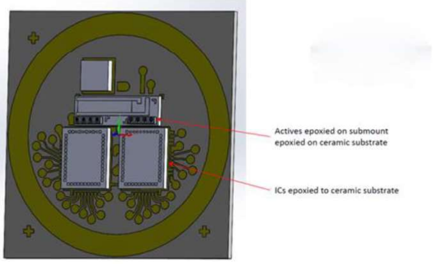

Actives epoxied on submount epoxied on ceramic substrate
ICs epoxied to ceramic substrate

Fig. 4. ICs on hermetic hybrid circuit. A window cap is soldered to the gold ring to hermetically seal the ICs on the substrate (not shown for clarity.)

The transceiver hybrid, depicted in Figure 4, can be configured for different operational modes. For space flight use it can be configured in a pin-strapped mode of operation to minimize radiation sensitivity by relying on internal default settings of the silicon-germanium (SiGe) vertical cavity surface emitting laser (VCSEL) driver and transimpedance amplifier (TIA)/Limiting amplifier integrated circuits (ICs). In this mode, the amplifier ICs operate independently using default settings and thermal compensation algorithms that do not require intervention from an external microprocessor. The other mode of operation is to employ an external microprocessor to control the driver and TIA/Limiting amplifier via an I2C serial interface.

This permits the use of a radiation-tolerant microprocessor to host the firmware to provide for full control of the transceiver unit.

C. *Optical connector and active alignment*

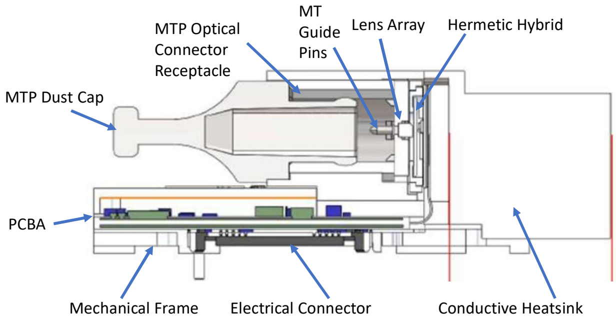


Fig. 5. Cross-section view of parallel optical transceiver construction.

A cross-sectional view of the transceiver is shown in Figure 5. The VCSEL and photodiode arrays are optically coupled directly to an MT connector interface using a molded glass aspheric lens array in an active optical alignment process. The coupling is in a linear optical path from laser/photodiode arrays to the MT contact fibre array, without any 90-degree mirror. This provides for very low-loss coupling, circularly-symmetric mode-filling of the fibre from the laser diode, and no over-filling of the photodiode active area which can lead to bandwidth limitation. Compared to competing parallel optical transceivers which utilize 45-degree angled coupling mechanisms and no lens arrays, the lensed approach leads to superior performance. This is especially evident at higher data rates of 28 Gbps, because all of the light from the optical fibre can be coupled to the active area of the photodiodes without overfilling, and all of the VCSEL output coupled to the optical fibre without vignetting, both of which can lead to bandwidth degradation.

Since the optical coupling to the MTP optical connector is non-contacting, there is no issue with vibration causing foreign objects debris (FOD) at the MT interface. The optical connector is easily inserted or removed, which simplifies installation of the transceiver onto the host PCB, or replacement of optical cabling in a chassis.

D. *Electrical high-speed connection*

The signals are routed from the hermetic hybrid through a flexible printed circuit section on matched impedance 100-ohm transmission lines to a rigid printed circuit board assembly (PCBA) in the bottom of the unit where all transmission lines go through decoupling capacitors and are then routed to a high-speed electrical board-to-board low-profile connector. This connector supports speeds up to 28 Gbps, and the inclusion of the decoupling capacitors in the unit reduces the footprint on the host PCBA.

The mating electrical connector is installed on the host PCB in in alignment with four solder-in threaded inserts. The transceiver unit has a rigid mechanical frame that is attached to the PCB using four captive screws. The mechanical frame removes all mechanical stresses from the electrical connector, and it therefore can withstand very high shock and vibration levels as well as repeated thermal cycling.

The electrical connector is capable of many mating cycles without degradation, with over 500 cycles demonstrated. This permits easy installation or replacement of a transceiver with minimal assembly time, compared to competing products requiring soldering of the transceiver to the host PCBA, or with excessively delicate electrical connectors.

E. *Mechanical and thermal construction*

The mechanical chassis of the unit provides a rigid assembly that protects the optoelectronic hybrid and supports the actively-aligned optical connector receptacle. This chassis supports the internal circuit board and also provides for highly-efficient heat-sinking of the opto-electronic hybrid directly to a massive heatsink. The heatsink is attached to the rear of the unit, opposite the optoelectronic hybrid. This heatsink can be either solid for conduction cooling in vacuum or finned for forced-air convection cooling in terrestrial or aircraft applications.

The typical thermal rise between the heatsink temperature and the internal temperature of the optoelectronic hermetic hybrid is less than 10 degrees C. Operation up to 100 C heatsink temperature has been demonstrated with good optical performance.

F. *Control options for thermal compensation, equalization and signal integrity in space environments*

The quad driver and TIA/Limiting amplifier ICs have extensive parameter settings accessible via the serial I2C bus, to adjust laser bias, equalization, pre-emphasis and other settings, as well as to support typical monitoring functions in the Digital Diagnostic Monitor Interface (DDMI) such as laser bias and received optical power. To achieve the highest level of data transmission performance over temperature, various parameters of these ICs must be adjusted as the device temperature varies to maintain an open data eye. Also, to improve the operation of devices in specific systems, equalization and pre-emphasis adjustments for each channel may be required to compensate for transmission line non-idealities or connectors, etc.

These adjustments are typically handled by a dedicated microprocessor. In terrestrial and aircraft versions of the product, a microprocessor IC is included in the unit to host the DDMI memory map. In “pin-strapped” mode, the user does not have I2C access to the registers on the VCSEL driver and TIA/limiting amplifier. In external-processor mode, an external microprocessor connects to the driver and limiting amplifier ICs via I2C bus. This provides the capability for the user’s radiation-tolerant external microprocessor to host the firmware that is resident in the onboard microprocessor of the terrestrial/aircraft version of the unit, which provides for advanced tuning, equalization, and temperature compensation to optimize signal integrity to the maximum extent possible.

G. *Design for radiation tolerance and low outgassing*

The SiGe driver and TIA/limiting amp receiver ICs are produced in 130nm SiGe process, and the laser diode and photodiode arrays are GaAs. These devices have been found to be quite radiation resistant (see data in later sections). However, in space applications with significant radiation exposure, the CMOS microprocessor is the weakest link. There is not room to include a radiation tolerant microprocessor, and it would add greatly to the cost of the unit. So for space applications, the microprocessor is deleted in favor of a user-supplied external microprocessor, or the driver and receiver ICs can be "pin-strapped" to operate from internal default settings.

When configured to operate with an external microprocessor or FPGA, this can provide similar control of the driver and receiver ICs. There is also provision for external control circuitry for the heater element. Alternatively, the driver and receiver ICs can operate without external control using default built-in parameters.

All epoxies and adhesives used in the transceiver have been verified to pass ASTM E595 outgassing testing.

III. Transceiver Test Data

A. Performance data

The parallel optical transceiver units meet the performance requirements as outlined in the SpaceFibre standard with margin, at data rates up to 14 Gbps.

As shown in Figure 6, the current consumption from a 3.3V supply of a transceiver module varies with temperature from 0 C to 85C, increasing from approximately 375 mA at 0 C to 400 mA at 85 C heatsink temperature. At temperatures below zero, when enabled, the heater circuit begins to draw current to keep the laser array temperature from falling below 0 C.

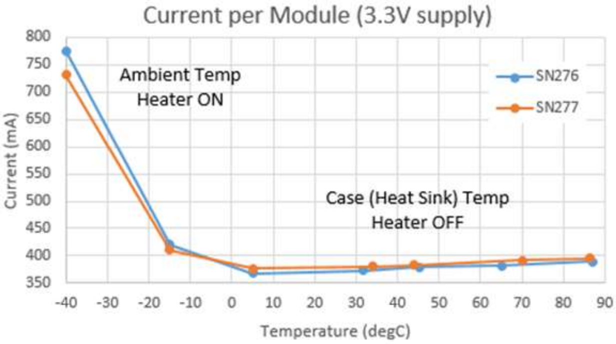
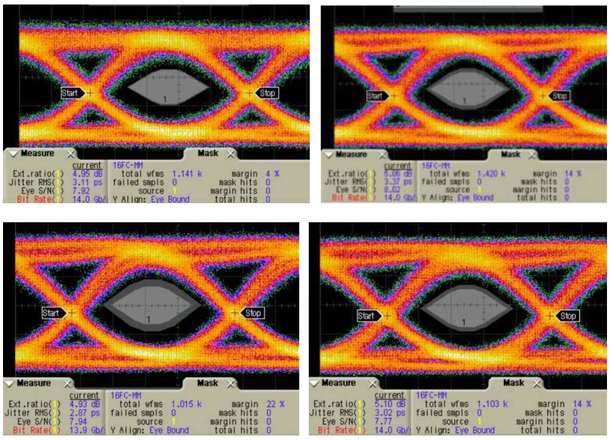


Fig. 6. Parallel optical transceiver current consumption vs temperature for two unit.

Transmitter eye measurements at 14 Gbps at -40C are shown in Figure 7 shows and Figure 8 shows the eye measurements at 14 Gbps at 85C. As can be seen, the transmitters have appreciable mask margin across the range -40 to +85C. The worst-case optical transmitter and receiver section performances at 85C heatsink temperature are summarized in Table I for a typical device. These values yield a range of optical link budget at 85C from minimum of 10.8 dB to maximum of 12.3 dB. At -40C, the optical power ranges from typical 1.5 – 2.0 dBm, and 1E-12 BER sensitivity of -12.9 to -13.2 dB, yielding link budget range at -40C of 14.4 to 15.2 dB. These performances were obtained with microprocessor control of the transceiver parameters optimized over temperature.



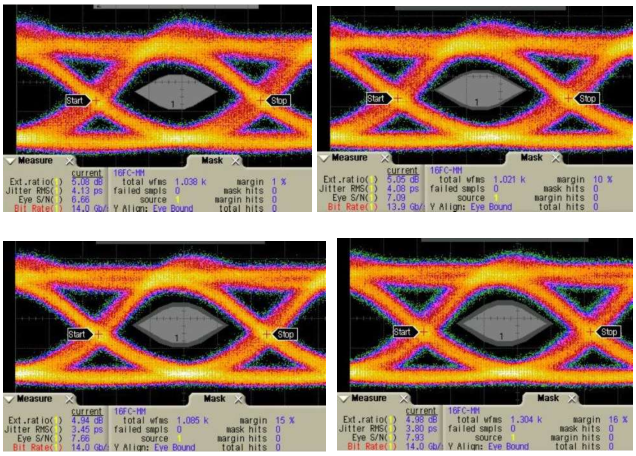
```
current          16FC-MM
Ext.ratio(1)  4.95 dB   total wfms   1.141 k   margin       4 %
Jitter RMS(1) 3.11 ps   failed smpls 0         mask hits    0
Eye S/N(1)    7.92      source       1         margin hits  0
Bit Rate(1)   14.0 Gb/s Y Align: Eye Bound     total hits   0
```

```
current          16FC-MM
Ext.ratio(1)  5.06 dB   total wfms   1.420 k   margin       14 %
Jitter RMS(1) 3.37 ps   failed smpls 0         mask hits    0
Eye S/N(1)    8.02      source       1         margin hits  0
Bit Rate(1)   14.0 Gb/s Y Align: Eye Bound     total hits   0
```

```
current          16FC-MM
Ext.ratio(1)  4.93 dB   total wfms   1.015 k   margin       22 %
Jitter RMS(1) 2.87 ps   failed smpls 0         mask hits    0
Eye S/N(1)    7.94      source       1         margin hits  0
Bit Rate(1)   13.9 Gb/s Y Align: Eye Bound     total hits   0
```

```
current          16FC-MM
Ext.ratio(1)  5.10 dB   total wfms   1.103 k   margin       14 %
Jitter RMS(1) 3.02 ps   failed smpls 0         mask hits    0
Eye S/N(1)    7.77      source       1         margin hits  0
Bit Rate(1)   14.0 Gb/s Y Align: Eye Bound     total hits   0
```

Fig. 7. Parallel optical transceiver transmitter eye at -40C.



```
current          16FC-MM
Ext.ratio(1)  5.08 dB   total wfms   1.038 k   margin       1 %
Jitter RMS(1) 4.13 ps   failed smpls 0         mask hits    0
Eye S/N(1)    6.66      source       1         margin hits  0
Bit Rate(1)   14.0 Gb/s Y Align: Eye Bound     total hits   0
```

```
current          16FC-MM
Ext.ratio(1)  5.05 dB   total wfms   1.021 k   margin       10 %
Jitter RMS(1) 4.08 ps   failed smpls 0         mask hits    0
Eye S/N(1)    7.09      source       1         margin hits  0
Bit Rate(1)   13.9 Gb/s Y Align: Eye Bound     total hits   0
```

```
current          16FC-MM
Ext.ratio(1)  4.94 dB   total wfms   1.085 k   margin       15 %
Jitter RMS(1) 3.45 ps   failed smpls 0         mask hits    0
Eye S/N(1)    7.66      source       1         margin hits  0
Bit Rate(1)   14.0 Gb/s Y Align: Eye Bound     total hits   0
```

```
current          16FC-MM
Ext.ratio(1)  4.98 dB   total wfms   1.304 k   margin       16 %
Jitter RMS(1) 3.80 ps   failed smpls 0         mask hits    0
Eye S/N(1)    7.93      source       1         margin hits  0
Bit Rate(1)   14.0 Gb/s Y Align: Eye Bound     total hits   0
```

Fig. 8. Parallel optical transceiver transmitter eye at 85 C.

TABLE I. TRANSMTTER AND RECEIVER DATA AT 85C

Data/Ch.	Ch1	Ch2	Ch3	Ch4
AOP (dBm)	-0.8	-0.1	-0.1	-0.1
ER (dB)	5.1	5.1	4.9	5.0
Jitter (ps_{rms})	4.1	4.1	3.5	3.8
SNR (dB)	6.7	7.1	7.7	7.9
Margin (%)	1	10	15	16

Data/Ch.	Ch1	Ch2	Ch3	Ch4
V_{pp} -3dBm	514	529	540	520
V_{pp} -10dBm	479	492	495	473
Jitter-3dBm	3.5	3.2	3.8	4.0
Jitter-10dBm	4.7	4.1	4.9	5.3
BER 1e^{-12} (dBm)	*-12.3*	*-12.4*	*-12.0*	*-11.6*

B. Environmental qualification testing

Multiple samples of transceivers mounted on evaluation boards were tested under avionic application requirements for Random Vibration, Mechanical Shock, Humidity, and other tests. All units tested passed all qualification tests.

The unit tested is representative of the PCB-mounted MTP-fibre products which include 4 channel transceivers, 12 channel transmitters, 12 channel receivers at both 10 Gbps and 28 Gbps. All parts use similarly-constructed and hermetically-sealed optical hybrid sub-assemblies. The 10 Gbps transceiver was selected for testing, since it is the first product in the portfolio developed and most mature.

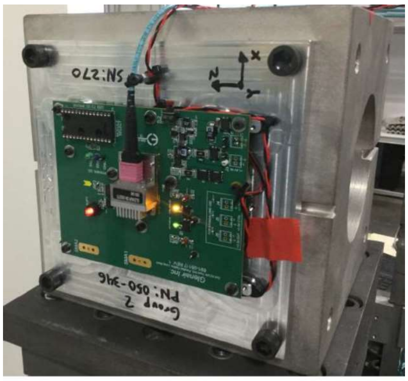


Fig. 9. Parallel optical transceiver on evaluation board, mounted on vibration and shock test fixture for operational loop-back test at 10 Gbps while undergoing shock and/or vibration exposure.

TABLE II. PARLLEL OPTICAL TRANSCEIVER QUALIFICATION TESTS

Test Item	Description	Reference	Group Number	Sampling (Qty.)
Random Vibration, Operating	Profile 46 grms 2 hours per axis: x, y & z	Mil-STD-810, Para. 514.6, proc. I SAE ARP6318 (Draft)	1	2 required 4 tested
Mechanical Shock, Operating	X- axis 650g 0.9ms, 10 pulses (5+ & 5-)	Mil-STD-810, Para. 516.6 SAE ARP6318 (Draft)	2	2
	Y- axis 650g 0.9ms, 10 pulses (5+ & 5-)	Mil-STD-810, Para. 516.6 SAE ARP6318 (Draft)		
	Z- axis 650g 0.9ms, 10 pulses (5+ & 5-)	Mil-STD-810, Para. 516.6 SAE ARP6318 (Draft)		
Temperature Cycling, Operating	100 cycles, -40ºC to +85ºC	ARINC 804-1 (MIL-STD-883H), Method 1010.8, Cond A. SAE ARP6318 (Draft)	3	2
Thermal Shock, Non-Operating	-55C and 125C, 500 Cycle	ARINC 804-1 (MIL-STD-883H), Method 1010.8, Cond B. SAE ARP6318 (Draft)	4	2
High Temperature Operating Life (Accelerated Aging, Operating)	1000 hours, +85ºC	ARINC 804-1 Section 4.9.6 SAE ARP6318 (Draft)	5	2 to 11 (2 only required per SAE)
ESD	500V HBV	ARINC 804-1 (MIL-STD-883H), Method 3015.8, Class 1C SAE ARP6318 (Draft)	6	1
Humidity, Operating (DC Power only)	10 days, RH 90% to 100%, Apply DC power only from step 1 to 6, then step 7 with subcycle DC disabled.	MIL-STD-883H, Method 1004.7 SAE ARP6318 (Draft)	7	2 required 4 tested
Fiber pigtail pull-test	Pull Test Force: 1 kg	Telcordia GR-468-CORE. Method 3.3.3.1.3	7	1

As shown in the summary Table II of results, several groups of identical production units built using the standard documented production process were subjected to 8 sets of tests. The units were powered on during exposures as indicated in the table. In the case of vibration, shock, thermal cycling and humidity, the units were passing 10 Gbps data on at least one channel, and monitored for error-free transmission.

The transceivers operated with zero bit errors prior to test, during test (if monitored) and after environmental exposure. Passing criteria were that the units operated within datasheet specifications with no bit errors before, during and after testing, and no change in optical output power of more than 0.5 dB from the power level measured prior to the environmental exposure.

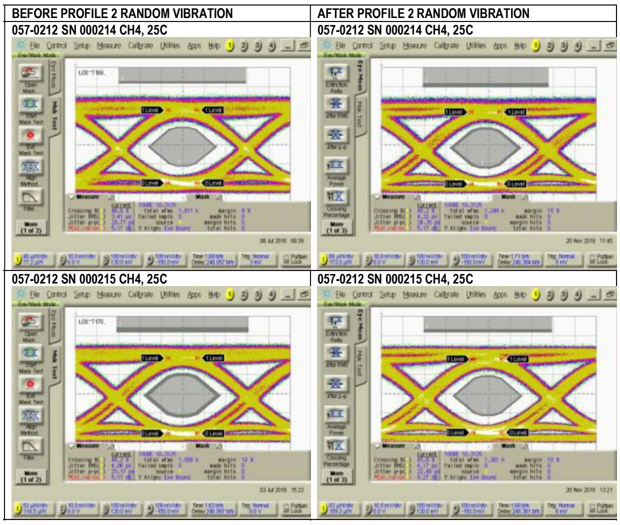

BEFORE PROFILE 2 RANDOM VIBRATION
AFTER PROFILE 2 RANDOM VIBRATION
057-0212 SN 000214 CH4, 25C
057-0212 SN 000214 CH4, 25C
057-0212 SN 000215 CH4, 25C
057-0212 SN 000215 CH4, 25C

Fig. 10. Representative transmitter optical eye diagrams before and after random vibration testing at 10.3125 Gbps.

C. *Transceiver radiation testing*

Radiation testing for proton, heavy ion and gamma exposures was performed with both pin-strapped (proton and gamma) and external processor (heavy ion and gamma) parallel optics transceivers. The primary purpose of these test campaigns was to obtain characterization data on the susceptibility of the devices to non-destructive single-event-effects (NDSEE) as well as determine if there were any proton or heavy ion induced destructive effects due to single-event-effects (SEE) or total ionizing dose (TID) to assess the suitability of these devices in radiation environments typical of spaceflight.

During the proton test three energies of protons were used to map out the energy vs upset cross-section curves for the observed NDSEE which included only single-event upsets (SEU) and no single-event functional interrupts (SEFIs). The term SEU is defined as a non-persistent data error as identified on the bit-error-rate test (BERT) equipment, while a SEFI is defined as persistent data corruption, generally requiring external mitigation such as a reset or power cycle.

Devices were irradiated with 250, 100 and 50 MeV surface energy protons with different flux intensities based on the Synchrotron capability. Fluxes at 50 and 100 MeV were ~4.6E+07 p/cm^2/s. At the higher energy, 250 MeV, the flux was ~2.3E8 p/cm^2/s. All runs at all energies were to a fluence of 1E+11 p/cm^2.

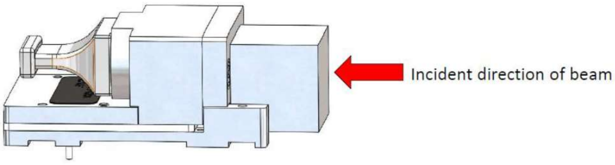


Fig. 11. Radiation beam direction for proton testing.

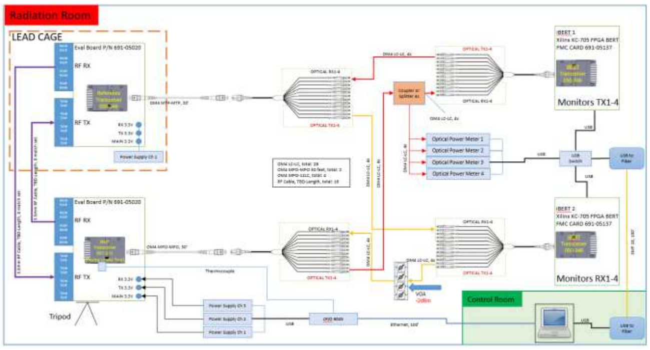
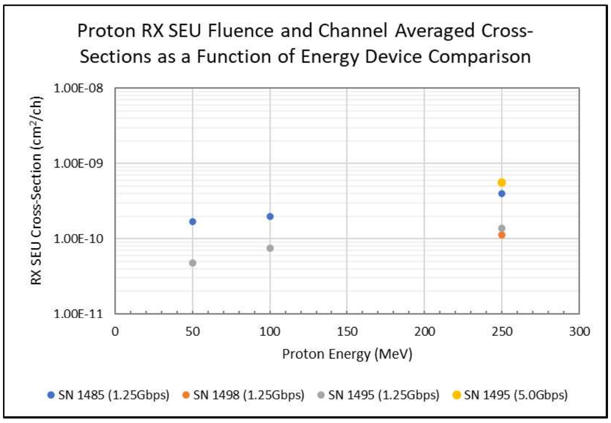


Fig. 12. Radiation test setup.



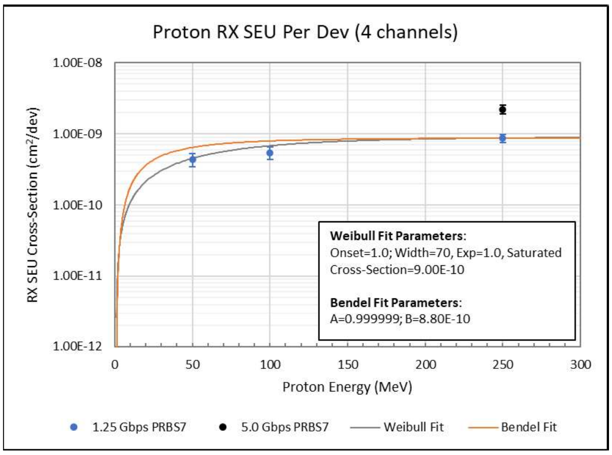


Fig. 13. Proton RX SEU cross-section plots with Weibull curve-fits.

The only observed SEE during proton testing were RX data errors. No TX data errors were observed and no SEFI on either RX or TX were observed to the highest fluence levels tested at all energies. Note that there were some small shifts in TX output power during exposures, but none resulted in a TX error. Based on the heavy ion SEE results, it is very likely that the proton fluences were insufficient to produce TX SEU and TX/RX SEFI. Heavy ion sensitivity would suggest that these events could be generated by secondary proton interactions if sufficient proton fluence was provided.

Additionally, proton specific TID testing was performed to verify functionality and parametric degradation up to 20 krad (Si). Both the transmitter (TX) and receiver (RX) section of a single device under test (DUT) were tested simultaneously, with independent data streams for the TX and RX sections using two BERT channels at 1.25 Gbps as the basis for most of the SEE testing. Incident optical power on the RX section of the DUT was set to approximately -2 dBm, which is well above the RX sensitivity threshold, to ensure that any bit errors detected were due to radiation-induced effects. 5 Gbps data rate was used during all biased TID testing as well as a small subset of SEE testing to evaluate the data rate impact on SEFI/SEU sensitivity. The data pattern used for all proton testing was PRBS7 due to test-set limitations. Temperature was monitored at the device heatsink during all exposures and was typically ~55C for biased exposures. All exposures were normal incidence of the heatsink relative to the beam.

In summary, all devices, biased and unbiased, survived proton TID exposure to 20 krad (Si) with negligible parametric drift.

Heavy ion Single Event Latchup (SEL) testing was conducted using Brookhaven National Lab (BNL)'s NASA Space Radiation Laboratory (NSRL) facility. For SEL testing we made use of the particles produced by BNL's Relativistic Heavy Ion Collider (RHIC). During the heavy ion test, five linear energy transfer (LET) values were used to similarly map out the SEU and SEFI upset cross-sections vs. LET.

Three transceiver devices were evaluated for destructive SEL at NSRL. The devices utilized external microcontrollers that were not in the ion beam, but at the end of a cable outside of the radiation exposure. Note that this contrasts with the proton testing in which pin-strapped devices were used without any microprocessor.

The DUT test fixture was mounted at normal incidence to the ion beam through the backside of the PCB. The DUTs were irradiated at elevated case temperature (~85 +/-5 C) and voltage (3.465V). The DUTs were irradiated with heavy ion particles at an effective LET value of ~45.68 MeV cm^2/mg. All three devices passed destructive SEL testing during all instances of irradiation.

Two transceivers with external microprocessor were also evaluated for Non-Destructive Single Event Effects (NDSEE) at NSRL. The devices utilized externally connected microcontrollers that were outside of the radiation beam. The DUT test fixture was mounted at normal incidence to the ion beam as was previously described for the proton and SEL results. The DUTs were irradiated at self-regulating case temperature of ~40C and 10% below operating voltage (3.135V) for worst case SEU conditions. Non-Destructive Single Event Effects (NDSEE) caused by ion strikes resulted in SEFI and SEU. Note that these results contrast with the proton test results as only RX SEU were observed with protons and both RX and TX SEU and SEFI were observed with the heavy ion testing.

On-orbit heavy ion Soft Error Rate (SER) estimates were calculated using the industry standard web-based CREME96 application. This application takes the Weibull parameter inputs and convolves them with the heavy ion LET spectra/ or proton spectra/distribution for the selected orbits, to estimate on-orbit SER for each particle type. For these example orbital SER calculations, a shielding thickness of 100 mils of aluminum was used to simulate a combined average thickness of the spacecraft enclosure.

SER are listed for select example orbital environments in Table III. Note that heavy ion only SER estimates have been calculated for both the expected SER (Fluence-Averaged) and as a worst-bounding-case SER (upper limit bound based on 95% confidence limits), where applicable. On-orbit SER estimates are provided per device for a single device for SEFIs. SER are listed for select example orbital environments.

TABLE III. CREME96 ON-ORBIT ERROR-RATE ESTIMATES

Environment	Polar Low Earth Orbit (700km 98.2° inclined circular orbit)		Equatorial Low Earth Orbit 1 (850km, 60° Inclined circular orbit)		Equatorial Low Earth Orbit 2 (1200km, 60° Inclined circular orbit)	
SEE Type	Expected Event Rate	Conservative (2σ) Event Rate	Expected Event Rate	Conservative (2σ) Event Rate	Expected Event Rate	Conservative (2σ) Event Rate
RX SEU (days/dev)	15	14	26	20	23	11
TX SBU (years/dev)	7.7	N/A	10.7	N/A	9.6	N/A
TX MBU (years/dev)	81.4	N/A	119.3	N/A	106.5	N/A
Register Error (years/dev)	8.1	N/A	12.9	N/A	11.5	N/A
Device SEFI (years/dev)	265.4	141.5	383.0	203.7	343.6	182.7

Environment	ISS		GEO		GPS – MEO (20,180KM 55° Inclined circular orbit)	
SEE Type	Expected Event Rate	Conservative (2σ) Event Rate	Expected Event Rate	Conservative (2σ) Event Rate	Expected Event Rate	Conservative (2σ) Event Rate
RX SEU (days/dev)	38	36	5	5	7	7
TX SBU (years/dev)	18.0	N/A	2.5	N/A	2.7	N/A
TX MBU (years/dev)	214.8	N/A	21.4	N/A	23.1	N/A
Register Error (years/dev)	23.0	N/A	2.4	N/A	2.6	N/A
Device SEFI (years/dev)	654.7	347.8	83.9	44.8	89.2	47.6

During the Co^{60} gamma testing, the devices were exposed up to 250 krad while operating, with no SEUs or SEFIs observed. Full operating performance was verified without degradation at the conclusion of the gamma exposures.

Overall, the radiation testing performed on these devices yielded favorable results and is indicative of potential space flightworthiness for select environments. Additional analysis of this information by system engineers/designers may be necessary to interpret the results for unique designs/space environments. Additional testing with heavy ions and potentially protons and gamma radiation may be warranted depending on program requirements and risk tolerance on a case-by-case basis.

IV. Parallel Optical Connectors

MT fibre optic ferrules are super-high-density commercial interconnect inserts that accommodate multiple rows of 12 fibres in a compact and lightweight format. MT ferrules can be terminated with a wide range of optical media including photonic flex circuitry, as well as ribbon and round cable. MT interconnects are typically used for backplanes or trunk lines — such as in spacercraft wiring — where one high density, multi-channel line feeds many branches. These industry-standard optical contacts are available in in physical-contact (PC), angle-polished-contact (APC) and expanded-beam types, for both single-mode and multi-mode optical fibre.

Ruggedized optical connectors in both circular and rectangular formats were developed and qualified for aerospace and military applications with PC, APC and expanded beam contacts. These connectors are ruggedized for aerospace applications and may be suitable for the fibreoptic physical layer of the SpaceFibre standard.

A. *Circular connectors*

Circular connectors in four shell-sizes and insert configurations were developed: single MT in size 11-1 arrangement, dual MT in 13-2 arrangement, three MTs in 15-3 arrangement, and four MT ferrules one size 17 shell in 17-4 arrangement. Since each ferrule can support two rows of 12 fibres, the 17-4 38999 connector shell supports up to 96 fibres in a single connector. These 38999 Mil-Spec MT solutions are ideally suited for commercial aircraft avionics, military / defense applications and other harsh-environment applications that require rugged MT performance.

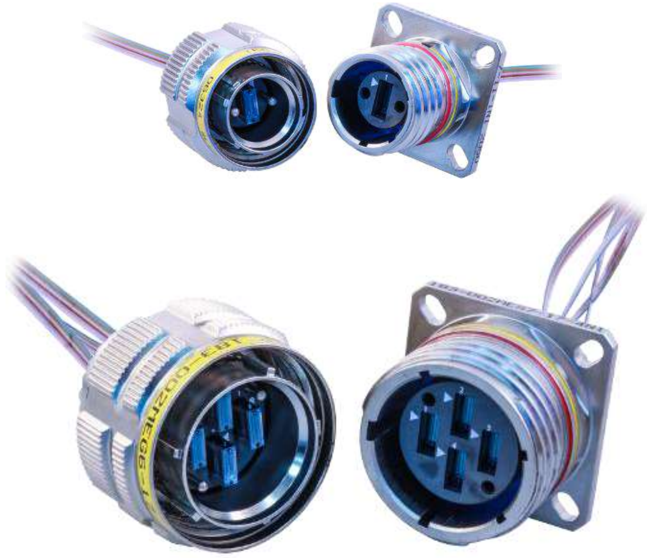


Fig. 14. Circular MT connectors: 11-1 arrangement (top) and 17-4 arrangement (bottom).

B. *Rectangular connectors*

MT fibre optic connectors in rectangular shells were also developed in three form-factors: single, dual and quad. These connectors are ideal for spacecraft applications where size and mass are critical, or panel space is at a premium such as in plug-in modules.

C. *Connector performance specifications summary*

The specifications for optical insertion loss and fibre type (in parentheses) are as follows:

Multimode Expanded Beam:	-0.5 dB Typical (50/125)
Multimode PC:	-0.3 dB Typical (50/125)
Singlemode PC:	-0.3 dB Typical (9/125)
Singlemode APC:	-0.3 dB Typical (9/125)

The specifications for Optical Back Reflection are as follows:

Multimode Expanded Beam	< -28 dB
Singlemode PC:	< -30 dB
Singlemode APC:	< -60 dB

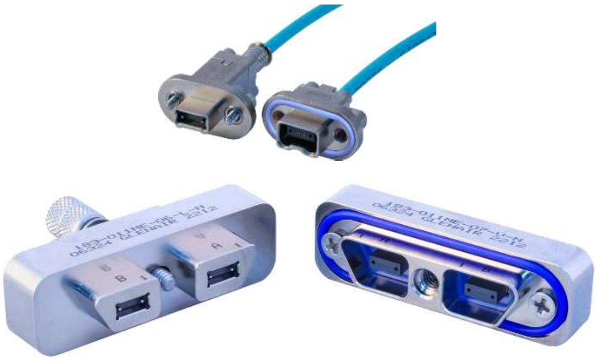


Fig. 15. Rectangular MT connectors: single-bay arrangement (top) and dual arrangement (bottom). Quad arrangement also available, not shown.

D. Qualification summary

Environmental qualification testing of the circular and rectangular MT fibre optic connectors was performed as summarized in Table IV. All connector types passed all qualification testing.

TABLE IV. CIRCULAR MT CONNECTOR QUALIFICATION TESTS

Test Parameter	Qualification Requirement
Mechanical Shock	300 G Half-sine Pulse, 3 ms Duration, 3 Times Both Direction Each Axis per TIA-455-14A
Vibration, Random	49.5 Grms at Ambient Temperature per MIL-STD-1678-3, Measurement 3201, Test Condition C, 5.3c, 8 hours exposure each axis
Mating Durability	500 Mating Cycles per TIA-455-21A
Humidity*	90%-95% RH, 96 hour Exposure per TIA-455-5C, Method A, Test Condition A *
Thermal Cycle*	5 Cycles, -40°C to 85°C with 1 hour Exposure per EIA-364-32F, Condition VIII, Method A
Temperature Life*	85°C for 336 hours per TIA-455-4C

* Cable and epoxy-dependent

V. SUMMARY AND CONCLUSION

SpaceFibre-compatible parallel optical transceivers and MT fibre optic connectors were developed and presented. Details of the design, performance, radiation testing and environmental qualification testing were presented. These results suggest very good potential for satisfying the requirements of the SpaceFibre standard for spacecraft applications.

ACKNOWLEDGMENTS

The authors acknowledge the contributions of their colleagues at Glenair in the development and testing of the hardware presented here (R. Wyss, M. Wabs, T. Sette, E. Chua, A. Chua, J. Delaney, K. Nguyen, I. Murillo, J. Padilla, R. Sawires, G. Noll, C. VanSoest) and at Troxel Aerospace (M. Gruber, J. Schaeffer and I. Troxel).

REFERENCES

[1] ESA SpaceFibre standard, ECSS-E-ST-50-11C, 2019.

[2] R.T. Logan Jr., “Photonic transceivers for SpaceFibre datalinks,” 8th InternaSpacewire Conference Proceedings, Long Beach, CA, May 2018.

Reliability Testing of 28Gbps/channel Fiber Optics Transceivers for Space Applications

Naeem Safdari
Opto-electronics Design Engineer
Smiths Interconnect Canada
Montreal, Canada
naeem.safdari@smithsinterconnect.com

Gabriel Monette
Design Engineering Lead
Smiths Interconnect Canada
Montreal, Canada
gabriel.monette@smithsinterconnect.com

Jade Beydoun
Engineering Test Technician
Smiths Interconnect Canada
Montreal, Canada
jade.beydoun@smithsinterconnect.com

Arlen Martin
Product Marketing Director, Optics
Smiths Interconnect Canada
Montreal, Canada
arlen.martin@smithsinterconnect.com

Robert Varano
Senior Principal Engineer
Smiths Interconnect Canada
Montreal, Canada
robert.varano@smithsinterconnect.com

Frédéric Laforce
Engineering Manager
Smiths Interconnect Canada
Montreal, Canada
frederic.laforce@smithsinterconnect.com

Abstract – Smiths Interconnect manufactures fiber optic multi-channel parallel optical transceivers. The transceiver product families consist of 4-channel and 12-channel versions with each channel capable of supporting data rates of up to 28Gbps independently. The optical interface is an integrated industry standard 1x12 MT ferrule optical fiber interface. The transceiver modules provide excellent optical intra-satellite high-speed communication links with a single +3.3V power supply and case operating temperature range from -40˚C to +85˚C. The devices are extremely light weight (3.0g), low power consumption and resistant to radiation effects and Electro-Magnetic Interferences (EMI). The transmitter channels are based on 850nm wavelength Vertical Cavity Surface Emitting Lasers (VCSELs) offering the best possible performance stability over a wide range of temperature without the need for power-hungry temperature controllers. In this talk, results from space and environmental qualification tests will be presented and more specifically, single event effect (heavy ions), total ionizing dose (gamma rays) and total non-ionizing dose (proton) radiations, live vacuum thermal cycling (TVAC) and outgassing. In addition to space qualification tests, further environmental qualification tests were completed to validate the mechanical integrity of the product, including live random vibration, mechanical shock, thermal shock, rapid decompression, temperature cycling, lifetime, and damp heat tests. The transceivers are built on the same manufacturing platform used for Smiths Interconnect's 10Gbps transceivers which are currently flying in space. Based on space qualification and environmental qualification test results, Smiths Interconnect's 28Gbps/channel transceivers have been proven to be well designed for the harsh space atmospheric environment and to be radiation tolerant.

***Keywords—Smiths Interconnect, Transceivers, Optical Module, Space*ABLE$^{®}$ 28.**

I. Introduction

Smiths Interconnect developed the *Space*ABLE$^{®}$28 LGA multi-channel parallel optical transceiver modules specifically for harsh space environment applications. The *Space*ABLE 28 product family is qualified for the space environment, passing extensive qualification tests. The modules prove their radiation resistance and mechanical integrity robustness. Optical fiber communication has proven to be the best technology choice for SpaceWire application and the Smiths Interconnect *Space*ABLE 28 LGA optical modules are very well suited for these applications, capable of sending and receiving optical data up to 28Gbps per channel over a case operating temperature range of -40 °C to +85 °C. The *Space*ABLE 28 optical modules are the most reliable optical modules for SpaceWire communication, where a huge amount of information may be sent and received within the satellite sub-systems over optical fiber rather than the heavy and less efficient copper alternative. The optical modules provide the best point-to-point optical data communication links up to 100 meters in length. These modules may be mounted mid-board or may also be mounted on the board edge, including Space VPX backplane applications. This report also presents the radiation, thermal and mechanical tests conditions and results.

II. Tranceivers Overview

A. Functionality

Smiths Interconnect *Space*ABLE 28 LGA optical modules are primarily a digital signal converter, converting electrical signals to optical and optical back to electrical signals. The four-channel transceivers (4TRX *Space*ABLE 28 LGA optical modules) are four-lane, full-duplex optical modules that include four optical transmit channels and four optical receive channels all in one small and ruggedized package (Fig. 1). The electrical interfaces are based on common mode logic (CML) (Fig. 2) and use 96-contact land grid array (LGA) interposers for the

Space Qualification is funded by EUROPEAN SPACE AGENCY (ESA)

electrical connection to a host board. Similarly, the 12-channel *Space*ABLE 28 LGA optical modules are half-duplex optical transmitters (12TX) or receivers (12RX), each in separate packages and capable of passing signals up to 28Gbps per channel.

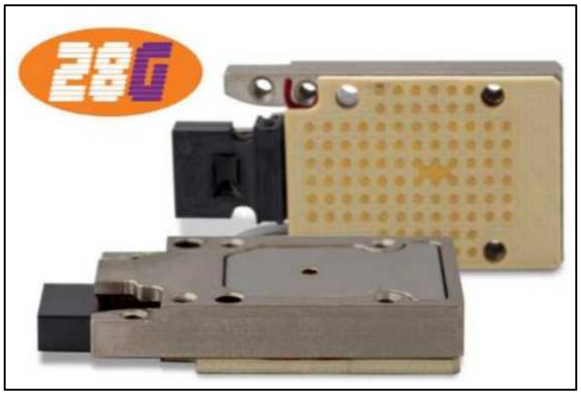


Fig. 1. *Space*ABLE® 28 LGA optical modules

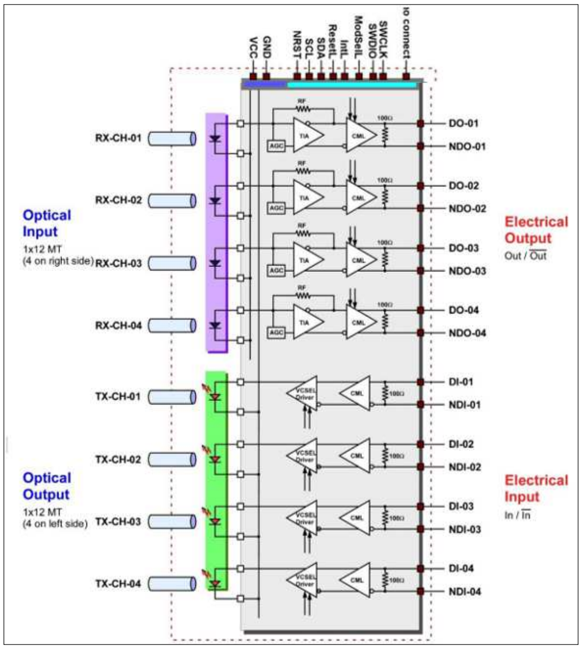


Fig. 2. Optical and CML Interfaces of a 4TRX Transceiver

B. *Compatibility*

The *Space*ABLE® 28 optical modules are ideal for mounting on densely populated boards where large amounts of data transfer are required. Applications involving processing, switching and more proprietary FPGA designs can benefit from the dense, high-speed, optical channels that provide much longer interconnect distances than copper links can. Applications like the aggregation and processing of very dense sensor information from high-definition cameras or phased-array radar sensors are ideally suited for the light-weight optical modules. In addition, the optical interface also allows easy fiber cable management and interoperability among a wide variety of modules, where each optical channel can also be operated independently (Fig. 3).

The *Space*ABLE 28 product family features:

- LGA interposer electrical connector
- Standard 1x12 multifiber termination (MT), optical interface
- Link distance of up to 100 meters with OM3 and OM4 fibers
- 850nm wavelength multimode light emitted from a vertical-cavity surface-emitting laser (VCSEL)
- 12 differential CML inputs or outputs
- NRZ type communication
- Asynchronous channel operation
- Data protocol-agnostic, balanced code
- Mid-board and edge-board mount configurations

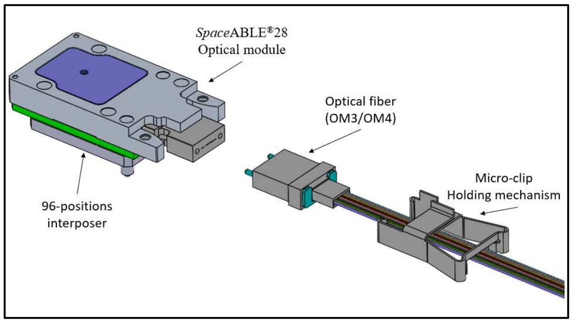


Fig. 3. Transceiver Optical Connection to Optical Fiber

C. *Physical Dimension and Configuration*

Being small, lightweight, and consuming less power are the key advantages of *Space*ABLE 28 optical modules for SpaceWire applications (Fig. 4).

- 4TRX Module size: 28.38 x 14.1 x 4.40 mm
  - o plus interposer height of 1.55 mm
- Weight: 3 grams
- Low power consumption: 0.314 watts per channel (12TX-12RX optical link)

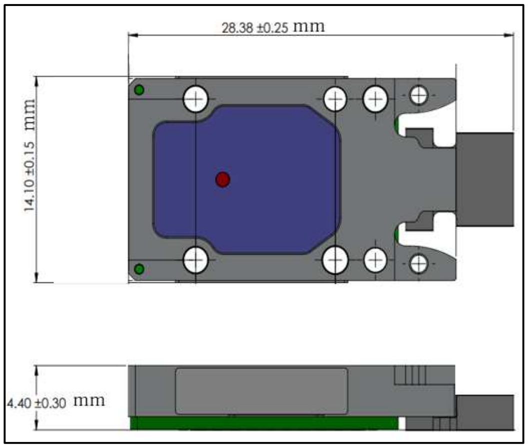


Fig. 4. Transceiver Dimensions

D. *Performance*

Table 1 presents a few of the basic specifications for the 12TX and 12RX optical performance, which is the key design strength of the optical modules.

TABLE 1. Transceiver Optical Performance

Parameter	Min	Typ	Max	Unit
Bit rate	1	25.78125	28.05	Gbps
Link budget margin	7			dB
Transmitter				
Avg optical power (per channel) at 25°C	2			dBm
Extinction ratio		5		dB
Center wavelength	840	850	860	nm
Receiver				
Sensitivity (per channel) at 25°C for BER 1E-09	-5			dBm
Optical power saturation limit	10			dBm
Peak sensitivity wavelength	840	850	860	nm

III. Radiation and Qualification Results

Radiation and qualification tests are conducted on the *Space*ABLE 28 product family to prove the level of radiation resistance and mechanical integrity robustness. The following sub-sections provide the results of the tests.

A. *Heavy Ions Radiation*

Heavy ions radiation causes Single Event Effects (SEE) on microelectronics in space environments, causing the microcircuits to malfunction by inducing soft errors or complete burnout of the device. As required for any microelectronic circuits, the optical modules must also be qualified for this space environment test.

The heavy ions radiation tests are conducted based on ESCC 25100, Issue 2 standard [2] and for real-time SEE measurement during heavy ions radiation. The optical modules were tested live at 25 °C and 85 °C case temperatures with pseudo-random binary sequence bit pattern (PRBS-31) at a rate of 25.87125Gbps running through all channels of the optical modules. Table 2 presents the list of heavy ions selected for radiation testing and the test conditions.

TABLE 2. Heavy Ions Radiation Test Conditions

	Device Under Test	Optical Module Type	Voltage (V)	Case Temperatures (°C)	(LET) (Mev.cm^2/mg)
1	LC0M0331	12TX	3.3	25	2.6, 8.0, 18.7, 40.3 and 66.7
2	LC0M0332	12TX	3.3 & 3.4	25 & 85	2.6, 8.0, 18.7, 40.3 and 66.7
3	LC0M0336	12TX	3.3	25	2.6, 8.0, 18.7, 40.3 and 66.7
4	LC0N0547	12RX	3.3 & 3.4	25 & 85	2.6, 8.0, 18.7, 40.3 and 66.7
5	LC0N0548	12RX	3.3	25	2.6, 8.0, 18.7, 40.3 and 66.7
6	LC0N0551	12RX	3.3	25	2.6, 8.0, 18.7, 40.3 and 66.7
7	LC0L0986	4TRX	3.3	25	2.6, 8.0, 18.7, 40.3 and 66.7
8	LC0L0987	4TRX	3.3	25	2.6, 8.0, 18.7, 40.3 and 66.7
9	LC0L0989	4TRX	3.3 & 3.4	25 & 85	2.6, 8.0, 18.7, 40.3 and 66.7

Heavy ions radiation test, additional conditions:

- Flux: 3.3x10^4 (ions/cm^{2}s)
- Total fluence: 1x10^7 (ions/cm^2)
- Radiation exposure time: 5 min
- Low Earth Orbit (LEO): 1200 Km
- Geostationary Orbit (GEO): 35786 Km
- Heavy ions:
  - Neon (Ne)
  - Argon (Ar)
  - Copper (Cu)
  - Silver (Ag)
  - Holmium (Ho)

Fig. 5 presents the heavy ions radiation test setup for four-channel optical modules, where the device under test (DUT) is mounted on a test board and the DUT is positioned 30 millimeters in front of the beam gun at a 90-degree angle. The DUT lid was removed and the beam was hitting the DUT chip perpendicularly. A reference station is used to send and receive a signal from the DUT and detect any soft errors caused by radiation. A temperature controller is used to heat-up the DUT for 85 °C testing during radiation. Both the reference station and the temperature controller are powered by two different power supplies that are connected to a laptop for data and power consumption recording.

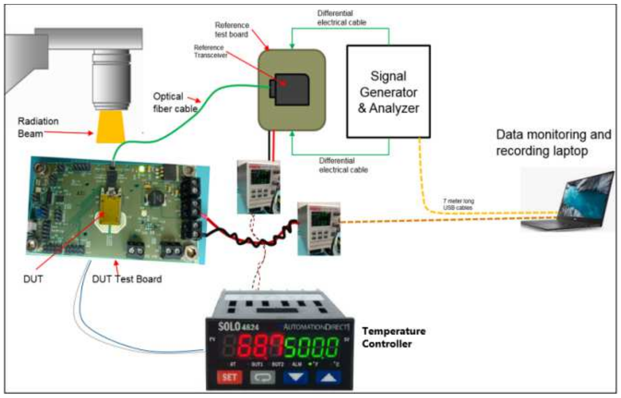


Figure 5. Heavy Ions Radiation Test Setup

There are three SEE definitions:

- Single Event Functional Interrupts (SEFI) is when the device under radiation loses functionality and goes to reset mode.
- Single Event Latch-up (SEL) is when the device under radiation malfunctions by going into another high-current consumption state.
- Single Event Upsets (SEU) is when the device under radiation is affected by heavy ions and causes soft errors of the data communication.

Fig. 6 presents a typical example of a *Space*ABLE 28 product's bit error rate (BER) live monitoring under radiation by heavy ions. The BER is real-time error acquisition during all selected heavy ions. The BER of channel 1 and channel 4 increased as the error count was gradually increasing for every ion. The BER of channel 2 and channel 4 spiked during BER logging that is an indication a burst of errors caused by radiation, but the DUT fully recovered within one second.

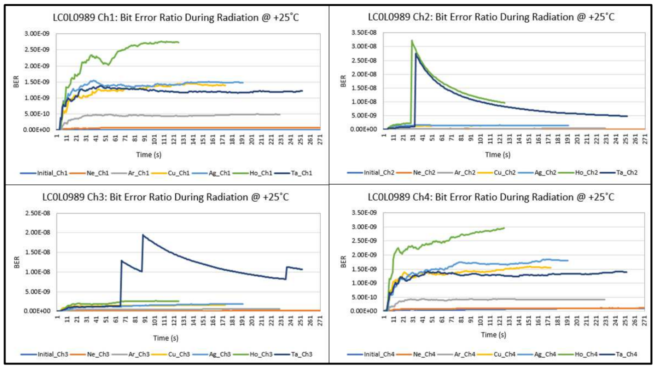


Fig. 6. Transceiver BER Monitoring Under Heavy Ions Radiation

None of the DUT exhibited SEL. All of the devices under radiation remained fully functional and had very stable power consumption Fig. 7.

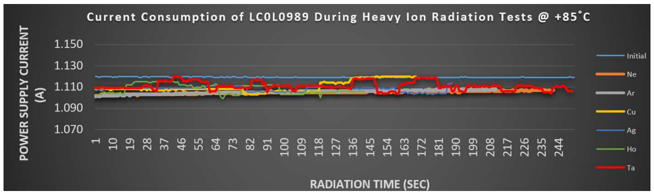


Fig. 7. Transceiver Power Consumption Under Radiation

Petersen's Figure of Merit (FOM) is used for the SEU rate calculation with following two equations [3]

$$FOM = \frac{\sigma_{HL}}{L_{0.25}^{2}} \left[\frac{(MeV/(mg{*}cm^2)^2}{cm^2}\right] \tag{1}$$

$$R = [C \; x \; FOM \quad] \tag{2}$$

Where:

- σ_{HL} is the limiting cross-section
- $L_{0.25}^{2}$ is the LET at ¼ of the saturation point
- R is the rate for a particular orbit
- C is coefficient rate for proton and heavy ion environment.

The cross-section is defined as the average number of errors over total fluence as shown in Fig. 8. The 4TRX cross-section curve is showed in this figure. 12TX and 12RX cross-section curves are used for rate calculation but not included in this report.

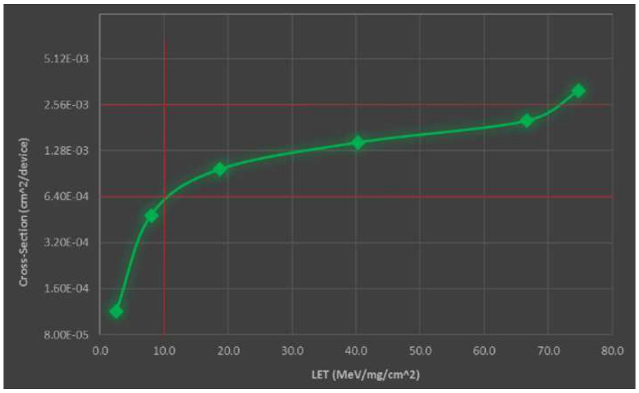


Fig. 8. 4TRX Heavy Ions Radiation Cross-Section vs LET

Tables 3, 4 and 5 contain 4TRX, 12TX and 12RX full SEU rate calculation and FOM parameters. The SEU rate calculation is done for shielded and unshielded devices. According to table 3, for a shielded device, the probability that one error or event can occur in a 4TRX optical module due to radiation is 0.00965 per day for Geostationary Orbit (GEO) and 0.0775 events per day for Low Earth Orbit (LEO) that is 3.52 (GEO) and 28.3 (LEO) events per year. An event is defined as one soft error in SEU rate calculation. The probability of one event per day for shielded devices is even much lower than for unshielded devices as shown in the tables below.

For calculating this rate, the worst-case altitude (1,200 Km) is considered for a LEO orbit and a 100 mils thick unshielded case is included in Petersen's model.

Table 3. 4TRX Figure of Merit Parameters

Parameters	Channel 1	Channel 2	Channel 3	Channel 4
Fluence (ions/cm^2)	10^7	10^7	10^7	10^7
σ_{HL}	1.14E-03	3.56E-03	3.20E-03	1.91E-03
$L_{0.25}$ (MeV*cm^2)/mg	8.0	15	9	8
FOM	1.78E-05	1.58E-05	3.95E-05	2.98E-05
C (Geo)	375	375	375	375
C (Leo)	3.01E+03	3.01E+03	3.01E+03	3.01E+03
Rate/day (Geo)	6.68E-03	5.93E-03	1.48E-02	1.12E-02
Rate/day (Leo)	5.36E-02	4.76E-02	1.19E-01	8.98E-02
With Shielding, t = Medium Thickness = 200 Mils				
C (Geo)	318.6	318.6	318.6	318.6
C (Leo)	2.56E+03	2.56E+03	2.56E+03	2.56E+03
Rate/day (Geo)	5.67E-03	5.04E-03	1.26E-02	9.51E-03
Rate/day (Leo)	4.55E-02	4.04E-02	1.01E-01	7.63E-02

Similarly, the SEU rate for 12TX and 12RX is shown in tables 4 and 5. The probability of getting one event per year is still very low for both 12TX and 12RX.

12TX SEU rates (unshielded) on GEO and LEO orbits are 0.97 and 7.79 event per year, respectively.

12RX SEU rates (unshielded) on GEO and LEO orbits are 6.9 and 554 event per year.

Table 4. 12TX Figure of Merit Parameters

σHL =	Saturation or Limiting Cross-Section		7.09E-04	
L0.25 =	LET at 25% of the Limiting Cross-Section		10	
Cross-Section	# Event / Fluence			
Fluence	10^7 (ions/cm^2)			
FOM	7.09E-06			
No shielding				
Rate Coefficient (Geo) =	3.75E+02			
Rate Coefficient (Leo) =	3.01E+03			
Rate (Geo) =	2.66E-03	event/day	9.70E-01	event/year
Rate (Leo) =	2.13E-02	event/day	7.79E+00	event/year
with 200 MILs thickness of shielding				
Rate Coefficient (Geo) =	3.19E+02			
Rate Coefficient (Leo) =	2.56E+03			
Rate (Geo) =	2.26E-03	event/day	8.24E-01	event/year
Rate (Leo) =	1.81E-02	event/day	6.61E+00	event/year

Table 5. 12RX Figure of Merit Parameters

σHL =	Saturation or Limiting Cross-Section		1.82E-02	
L0.25 =	LET at 25% of the Limiting Cross-Section		6	
Cross-Section	# Event / Fluence			
Fluence	10^7 (ions/cm^2)			
FOM	5.04E-04			
No shielding				
Rate Coefficient (Geo) =	3.75E+02			
Rate Coefficient (Leo) =	3.01E+03			
Rate (Geo) =	1.89E-01	event/day	6.90E+01	event/year
Rate (Leo) =	1.52E+00	event/day	5.54E+02	event/year
with 200 MILs thickness of shielding				
Rate Coefficient (Geo) =	3.19E+02			
Rate Coefficient (Leo) =	2.56E+03			
Rate (Geo) =	1.61E-01	event/day	5.86E+01	event/year
Rate (Leo) =	1.29E+00	event/day	4.71E+02	event/year

B. *Total Ionization Dose*

Total Ionization Dose (TID) tests from Cobalt-60 gamma ray radiations are done to emulate the presence and the impact of such radiation on high-speed opto-electronics circuits when used for space applications such as intra-satellite communications. TID tests are conducted based on ESCC 22900, Issue 5 standard with following conditions:

- Dose rate: 100 rad/h
- Dose levels: 0, 25, 51, 76 and 107 kilorads
- Post irradiation annealing for 24 hours at +25°C
- Post irradiation annealing for 168 hours at +100°C

The 12TX and 12RX TID radiation and interim tests sequence is presented in Fig. 9.

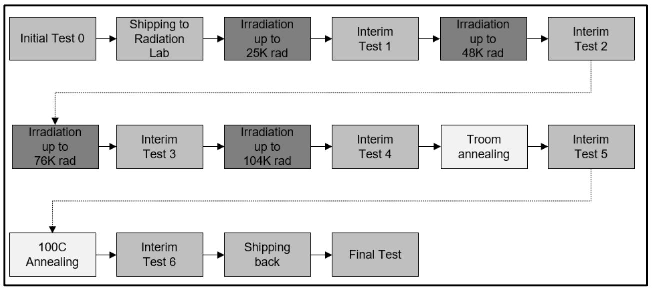


Fig. 9. Total Ionization Dose Radiation and Test Sequence

An interim test is done after each radiation level as presented in Fig. 9. Eight units are prepared for TID radiation, four biased and four unbiased while exposed to radiation. Table 6 presents the biased units interim test results, and each column of that table is showing the number of errors after each interim test. Both biased and non-biased units resulted in errors on some channels after the first step of radiation but the errors cleared after annealing. An increasing number of errors shows the affect of total ionization irradiation on optical modules and proving that the units survived Cobald-60 irradiation without any permanent damage.

Table 6. TID Biased Units Interim Tests

Un-biased Pairs	Channels	Pretest at TRAD	Interim Test After 25K	Interim Test After 48K	Interim Test After 76K	Interim Test After 104K	Interim Test After 24h Annealing	Interim Test After 168h Annealing
LC0M0362 - LC0N0494	ch1	0	0	0	0	0	1	0
	ch2	0	0	0	0	0	0	0
	ch3	0	0	0	0	0	0	0
	ch4	1	0	0	0	1168835	0	0
	ch5	40	11693	835	0	2439	0	0
	ch6	2265	11491	18786	1094	9	4	0
	ch7	0	12139	94	335	1	0	1
	ch8	2	204	61	34550	14	22	113
	ch10	0	1	0	0	0	67	0
LC0M0364 - LC0N0521	ch1	0	0	0	0	409	0	0
	ch2	0	0	0	0	0	0	0
	ch3	0	0	0	0	0	0	0
	ch4	0	0	0	0	0	207002832	0
	ch5	0	0	0	0	1	17	30
	ch6	8568	32767	102948	76359	735	349	498
	ch7	11	3851	0	300	0	520	54
	ch8	1	0	0	0	0	0	0
	ch10	1	1110	1	3	0	26	0
	ch11	0	9	286	0	2	4	0
	ch12	0	0	0	0	0	0	0

Table 7. TID Un-Biased Units Interim Test

Un-biased Pairs	Channels	Pretest at TRAD	Interim Test After 25K	Interim Test After 48K	Interim Test After 76K	Interim Test After 104K	Interim Test After 24h Annealing	Interim Test After 168h Annealing
LC0M0362 - LC0N0494	ch1	0	0	0	0	0	1	0
	ch2	0	0	0	0	0	0	0
	ch3	0	0	0	0	0	0	0
	ch4	1	0	0	0	1168835	0	0
	ch5	40	11693	835	0	2439	0	0
	ch6	2265	11491	18786	1094	9	4	0
	ch7	0	12139	94	335	1	0	1
	ch8	2	204	61	34550	14	22	113
	ch10	0	1	0	0	0	67	0
LC0M0364 - LC0N0521	ch1	0	0	0	0	409	0	0
	ch2	0	0	0	0	0	0	0
	ch3	0	0	0	0	0	0	0
	ch4	0	0	0	0	0	207002832	0
	ch5	0	0	0	0	1	17	30
	ch6	8568	32767	102948	76359	735	349	498
	ch7	11	3851	0	300	0	520	54
	ch8	1	0	0	0	0	0	0
	ch10	1	1110	1	3	0	26	0
	ch11	0	9	286	0	2	4	0
	ch12	0	0	0	0	0	0	0

C. Total Non-Ionization Dose

The purpose of Total Non-Ionization Dose (TNID) radiation test is to confirm the radiation hardness level in a proton radiation environment. TNID radiation tests are needed to emulate the presence and impact of such radiations on high-speed opto-electronics circuits when used for space applications such as SpaceWire and intra-satellite communications. TNID radiation conditions are as following:

- Proton beam energy: 100 MeV
- Dose Levels: 5x10^{5}, 5x10^{11} and 5x10^{12} protons/cm^{2}
- Post irradiation annealing at 100°C for 168 hours

Table 8 present the TNID devices under test (DUT) and level of radiation dose for each (DUT).

TABLE 8. TNID Optical Modules and Dose Levels

12TX	12RX	**Radiation Levels** protons/cm^{2}
LC0M0479	LC0N0595	5E+10
LC0M0467	LC0N0522	5E+10
LC0M0445	LC0N0531	5E+10
LC0M0450	LC0N0558	5E+11
LC0M0448	LC0N0529	5E+11
LC0M0447	LC0N0536	5E+11
LC0M0435	LC0N0592	5E+12
LC0M0478	LC0N0591	5E+12
LC0M0342	LC0N0578	5E+12

All DUTs are tested before and after TNID radiation as initial and final tests, both test results are presented below in Table 9. According to the data, all DUTs have passed the performance test and the total non-ionization dose test without any permanent damage or performance degradation.

TABLE 9. TNID Units Performance Results

Total Fluence of 5E+10 p/cm2					Total Fluence of 5E+11 p/cm2					Total Fluence of 5E+12 p/cm2				
	channels	Sensitivity Level (dBm)	Initial Error Count	Final Error Count		channels	Sensitivity Level (dBm)	Initial Error Count	Final Error Count		channels	Sensitivity Level (dBm)	Initial Error Count	Final Error Count
Reference Tx			LC0M0445	LC0M0450	Reference Tx			LC0M0445	LC0M0479	Reference Tx			LC0M0439	LC0M0479
LC0N0522	Ch1	-5	413	0	LC0N0529	Ch1	-5	413	0	LC0N0578	Ch1	-5	0	1
	Ch2	-5	65	0		Ch2	-5	65	0		Ch2	-5	2	0
	Ch3	-5	211	4		Ch3	-5	211	0		Ch3	-5	0	0
	Ch4	-5	1	0		Ch4	-5	1	0		Ch4	-5	87	0
	Ch5	-5	2	0		Ch5	-5	2	0		Ch5	-5	0	1
	Ch6	-5	2	0		Ch6	-5	2	0		Ch6	-5	4	0
	Ch7	-5	0	0		Ch7	-5	0	21		Ch7	-5	0	0
	Ch8	-5	0	0		Ch8	-5	0	0		Ch8	-5	0	0
	Ch9	-5	0	2		Ch9	-5	0	0		Ch9	-5	0	0
	Ch10	-5	467	0		Ch10	-5	467	0		Ch10	-5	2	0
	Ch11	-5	1	64		Ch11	-5	1	0		Ch11	-5	2	0
	Ch12	-5	25	112		Ch12	-5	25	0		Ch12	-5	0	0
Reference Tx			LC0M0442	LC0M0479	Reference Tx			LC0M0478	LC0M0450	Reference Tx			LC0M0435	LC0M0450
LC0N0531	Ch1	-5	0	0	LC0N0536	Ch1	-5	0	0	LC0N0591	Ch1	-5	1535	0
	Ch2	-5	0	0		Ch2	-5	0	0		Ch2	-5	792	0
	Ch3	-5	0	0		Ch3	-5	0	0		Ch3	-5	8	3
	Ch4	-5	0	0		Ch4	-5	0	0		Ch4	-5	0	0
	Ch5	-5	1	0		Ch5	-5	9	0		Ch5	-5	14	0
	Ch6	-5	0	0		Ch6	-5	0	0		Ch6	-5	0	0
	Ch7	-5	0	0		Ch7	-5	0	0		Ch7	-5	0	0
	Ch8	-5	0	0		Ch8	-5	0	0		Ch8	-5	1	1
	Ch9	-5	0	0		Ch9	-5	0	15		Ch9	-5	0	0
	Ch10	-5	434	0		Ch10	-5	303	9		Ch10	-5	0	2
	Ch11	-5	0	0		Ch11	-5	41	3		Ch11	-5	0	100
	Ch12	-5	1.64E+12	0		Ch12	-5	31	83		Ch12	-5	3	384
Reference Tx			LC0M0479	LC0M0450						Reference Tx			LC0M0457	LC0M0479
LC0N0595	Ch1	-5	0	0	LC0N0558	Ch1				LC0N0592	Ch1	-5	0	0
	Ch2	-5	0	0		Ch2					Ch2	-5	4	0
	Ch3	-5	0	0		Ch3					Ch3	-5	88	0
	Ch4	-5	0	0		Ch4					Ch4	-5	0	0
	Ch5	-5	0	0		Ch5					Ch5	-5	0	0
	Ch6	-5	0	0		Ch6					Ch6	-5	425	0
	Ch7	-5	0	0		Ch7					Ch7	-5	0	0
	Ch8	-5	0	0		Ch8					Ch8	-5	47	11
	Ch9	-5	18	0		Ch9					Ch9	-5	0	0
	Ch10	-5	0	0		Ch10					Ch10	-5	0	0
	Ch11	-5	0	2		Ch11					Ch11	-5	0	0
	Ch12	-5	0	0		Ch12					Ch12	-5	0	0

IV. Outgassing Test Results

The outgassing test determines the ability of optical modules to operate in a vacuum in space environment without contaminating other nearby elements of the sub-system. Outgassing testing is done in accordance with ECSS-Q-ST-70-02C with the following pass criteria:

- Recovered Mass Loss (RML) < 1.00%
- Collected Volatile Condensable Material (CVCM) < 0.10%.

The outgassing test is done on 4TRX, 12TX modules and a 96-position interposer presented in Fig. 10. The 12TX optical modules are used as a test vehicle for qualification of the 12RX modules because they both have identical materials in their assemblies.

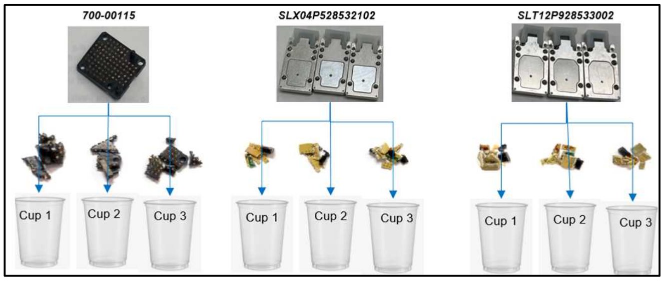


Fig. 10. Outgassing Test Samples

Table 10 presents the outgassing outcome. All three products successfully passed the standard pass/fail criteria for this test.

Table 10. Outgassing Test Results

1X	1.55 mm 96pos 13743 Interposer			3X	SLX04P528532102			3X	SLT12P928533002		
	TML %	RML %	CVCM %		TML %	RML %	CVCM %		TML %	RML %	CVCM %
Cup 1	0.077	0.052	0.002	Cup 1	0.215	0.189	0.031	Cup 1	0.158	0.124	0.029
Cup 2	0.073	0.052	0.003	Cup 2	0.206	0.179	0.044	Cup 2	0.136	0.094	0.018
Cup 3	0.064	0.051	IR plate	Cup 3	0.204	0.177	IR plate	Cup 3	0.123	0.085	IR plate
Average	0.071	0.052	0.002	Average	0.208	0.182	0.038	Average	0.139	0.101	0.024
Passing Limits	Not defined	< 1.00	<0.10	Passing Limits	Not defined	< 1.00	<0.10	Passing Limits	Not defined	< 1.00	<0.10

V. Thermal Test Results

A. Thermal Vacuum Test

Thermal Vacuum (TVAC) Test specifications are the following:

- Vacuum less than 5x10^{-5}hPa
- 20 temperature cycles from -40 °C to 85 °C with +/-5 °C precision.
- 5 minutes of dwell time at -40°C and +85 °C.
- Temperature ramp rate of less than 10 °C/min
- Live Bit Error Rate (BER) monitoring.

TVAC test setup and results of 4TRX are shown in this section. Three 4TRX were selected as DUT for TVAC and one of the three units was live tested whereas the other two were only biased (powered on) during TVAC test.

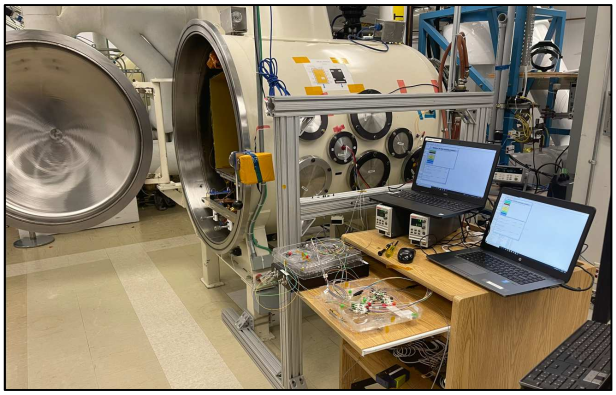


Figure 11. TVAC Testing Chamber and Test Station

The air pressure and temperature cycling are presented in Fig. 12 and Fig. 13. The vacuum level was monitored throughout the tests, as shown in Fig. 12. The lowest vacuum level was obtained during the first cycle at high temperature and reached below 10^{-5}hPa, better than our specified limit. The average vacuum level was around 5x10-7hPa.

The BER was verified to remain under 10^{-12} throughout the live TVAC test. All TVAC units were tested after TVAC test and all units passed the performance test.

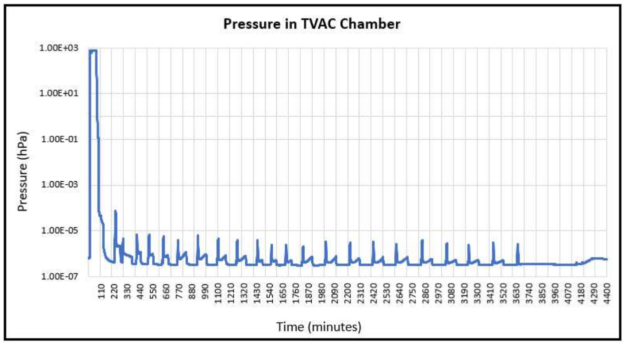


Fig. 12. Vacuum Pressure of TVAC Test

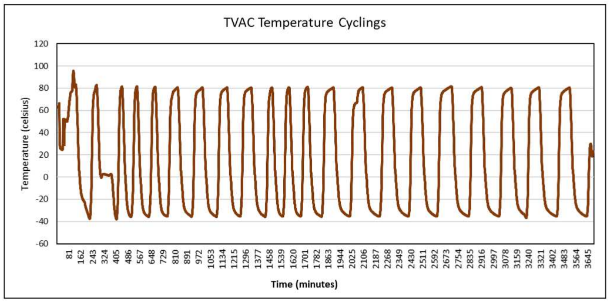


Fig. 13. TVAC Temperature Cycling

VI. Mechanical Test Results

A. *Random Vibration and Mechanical Shock Result*

Random vibration and mechanical shock tests are done on Smiths Interconnect's *Space*ABLE 28 products to ensure the mechanical robustness level of the optical modules. Random vibration and mechanical shock are needed as the units may undergo vibration and shock during spacecraft launch. The random vibration testing is done first and then followed by the mechanical shock test. An interim and an external visual inspection test are done after each test. All the devices under test were non-operational during the vibration and mechanical shock test, and the same devices were used for both tests.

The random vibration testing was done in accordance with MIL-STD-883, TM 2007 12 with 28.4 Grms perpendicular and 27.1 Grms parallel accelerations. Fig 14 presents the random vibration test setup.

The mechanical shock testing was done in accordance with MIL-STD-883 TM 2002 with 1500g acceleration, 0.5ms pulse width half-sine on all directions. Fig. 15 presents the mechanical shock test setup.

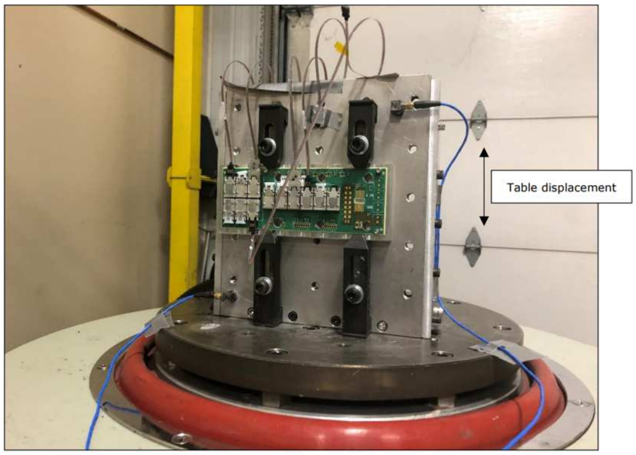


Fig. 14. Random Vibration Test Setup

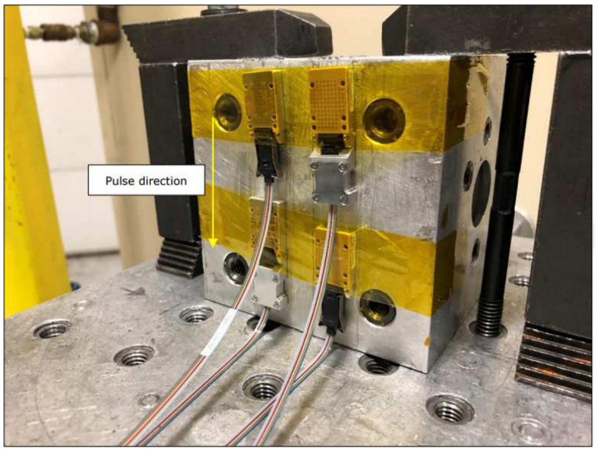


Fig. 15. Mechanical Shock Test Setup

The random vibration and mechanical shock units were tested before and after the random vibration and mechanical shock tests. The performance test results are shown in figures 16, 17 and 18. The post test results in the figures below prove that all DUT passed both random vibration and mechanical shock testing without any damage.

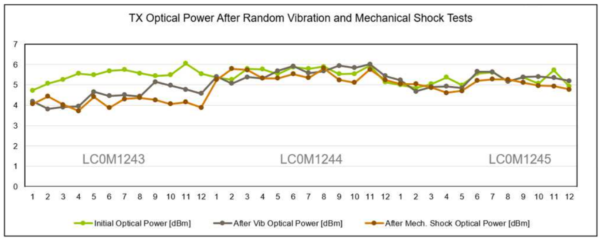


Fig. 16. 12TX Optical Power

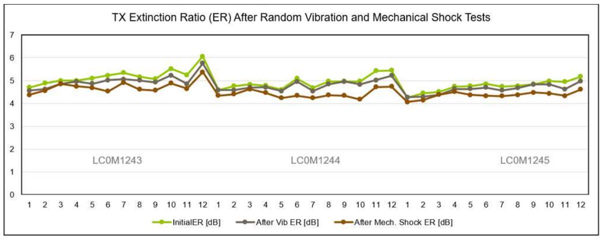


Fig. 17. 12TX Extinction Ratio

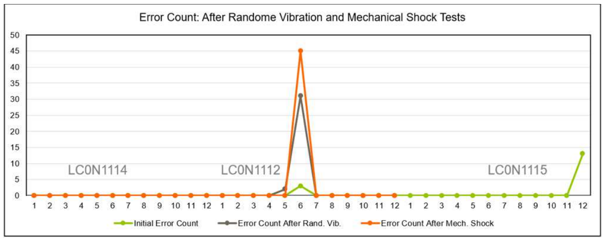


Fig. 18. 12RX Error Count Test Results

CONCLUSIONS

Optical communication through fiber optic for intra-satellite applications is an absolute requirement and the highly rugged and reliable *Space*ABLE 28 product family using parallel optics over OM3/OM4 multimode fiber provides the best data transfer service for SpaceWire and space optical communication. The *Space*ABLE 28 product family offers the best performance for any mid-board or edge-board mount configuration and passes both radiation and environmental qualification tests. The parts are available as Commercial Off-the-Shelf (COTS) product.

ACKNOWLEDGMENT

We gratefully acknowledge the technical support of the environmental, TID and TNID radiation lab staff.

REFERENCES

[1] Smiths Interconnect - Optical Embedded transceivers and modules

[2] E.L. Petersen, “SEE Rate Calculations Using The effective Flux Approach and a Generalized Figure of Merit Approximation”, *IEEE Trans. Nucl. Sci*., Vol. NS-42, pp1995-2003, 19

[3] JESD57A “Test Procedures for the Measurement of Single-Event Effects in Semiconductor Devices from Heavy Ion Irradiation” JEDEC Solid State Technology Association 2017, Sept., 2003.

SpaceFibre IP Cores for the Next Generation of Radiation-Tolerant FPGAs

Alberto Gonzalez Villafranca
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
alberto.gonzalez@star-dundee.com

Albert Ferrer Florit
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
albert.ferrer@star-dundee.com

Marti Farras Casas
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
marti.farras@star-dundee.com

Steve Parkes
STAR-Dundee
Dundee, Scotland, UK
steve.parkes@star-dundee.com

***Abstract*—SpaceFibre (ECSS-E-ST-50-11C) is a very high-performance, high-reliability and high-availability network technology specifically designed to meet the needs of space applications. It provides point-to-point and networked interconnections at Gigabit rates with Quality of Service (QoS) and Fault Detection, Isolation and Recovery (FDIR). SpaceFibre has been designed as a replacement of SpaceWire (ECSS-E-ST-50-12C)—it is backwards compatible with SpaceWire at the packet level—for next-generation space missions where very high throughput is required, providing up to 6.25 Gbit/s per lane, with multi-lane allowing to reach up to 16 times the speed of a single lane. NORBY and OPS-SAT technology demonstrators have already flown SpaceFibre, with more missions in both Europe and the USA currently designing or planning to use SpaceFibre.**

STAR-Dundee has developed a complete family of SpaceFibre IP cores fully compliant with the SpaceFibre standard. This family is composed of four different IPs: Single-Lane Interface, Multi-Lane Interface, Single-Lane Routing Switch and Multi-Lane Routing Switch.

A new generation of radiation-tolerant FPGAs is emerging to cope with the ever-growing processing power required by newer missions. Microchip has released the PolarFire RTPF500, Xilinx the Versal XQRVC1902, and NanoXplore the BRAVE NG-Ultra. SpaceFibre operation requires serial transceivers, which are already inbuilt in modern FPGAs. The IPs have been adapted to take advantage of the specific transceivers and memory blocks offered by these new FPGAs.

In this work we analyse in detail the performance of STAR-Dundee SpaceFibre IP cores on this new generation of FPGAs considering several performance metrics, e.g. maximum lane speed, resource usage, etc. We also compare the performance of the IPs with current state-of-the-art space-grade FPGAs, i.e. Microchip RTG4 and Xilinx Kintex UltraScale XQRKU060. This analysis can also be used as a representative benchmark to compare the performances of the different FPGAs available for space.

Keywords—SpaceFibre, Interface, Routing Switch, IP Cores, PolarFire RTPF500T, Versal XQRVC1902, BRAVE, NG-Large, NG-Ultra, RTG4, XQRKU060

I. Introduction

SpaceFibre (SpFi) [1] is a communication technology for use onboard spacecraft which was released as an ECSS standard in 2019 (ECSS-E-ST-50-11C). It provides point-to-point and networked interconnections at Gigabit rates while offering QoS and FDIR capabilities. SpFi interoperates seamlessly with a SpaceWire (SpW) [2] network over virtual channels (VCs) as it uses the same data packet definition. Furthermore, SpFi provides broadcast capabilities and can operate over either copper or fibre optic cables. To enhance throughput and robustness, SpFi links can also operate as multi-lane, thus allowing data of a single logical link to be spread over several individual physical lanes. This multi-lane operation provides higher data rates through lane aggregation, supporting any number of lanes (up to 16) and unidirectional operation. This effectively multiplies the throughput of the interface by combining several lanes into a link. Furthermore, when a lane fails the multi-lane mechanism supports hot and warm redundancy and graceful degradation by automatically spreading traffic over the remaining working lanes.

The Network layer in SpFi is responsible for transferring data packets over a link or network. The information to be sent uses the SpW format: <Destination Address> <Cargo> <End of Packet Marker>. The routing concepts are the same as in SpW including both path and logical addressing. The Network layer includes the definition of Virtual Networks (VN). These VNs are built from the interconnection between VCs of different ports. VNs enable the creation of flexible SpFi routing switches (also known as Routers) comprising SpFi interfaces and a fully configurable, non-blocking, high performance, routing switch. This routing switch typically supports up to 64 VNs, each VN effectively behaving like independent SpW networks capable of working at multi-Gbps rates.

STAR-Dundee has developed a range of SpFi IP cores compliant with the standard. The range is composed of four different IPs: Single-Lane Interface (SL Intf), Multi-Lane Interface (ML Intf), Single-Lane Router (SL Router) and Multi-Lane Router (ML Router). These IPs have been optimised for speed considering the timing constraints of the slower FPGAs for space. The family of SpFi IPs is also compatible with commercial FPGAs such as Microchip SmartFusion2 and PolarFire, or Xilinx 7-series, UltraScale, Versal, etc. The SL Intf IP has already been tested in orbit in two demonstrator missions: NORBY and OPS-SAT [3]. These collaborations have demonstrated operational SpFi links in space, thus providing fly heritage for this technology.

This paper is a follow-up from a previous paper presented in the 2018 International SpW Conference which analysed the existing SpFi IP cores performance in the state-of-the-art FPGAs at the time, the Microchip RTG4 and the Xilinx Virtex5-QV [4]. For this new work the ML Router IP has been added to the IP analysis. On the other hand, a new generation of radiation-tolerant FPGAs has emerged to cope with the growing processing power required by newer missions since the paper was published in 2018. The RTG4 has been kept in the analysis as reference for non-volatile legacy FPGA. The volatile reference has been updated from Virtex5-QV to the newer and more powerful Xilinx XQRKU060.

This paper describes the new generation of space-qualified FPGAs in section II. Section III analyses the main features of the STAR-Dundee SpFi IP Cores. Sections IV, V, VI and VII focus on the specific features and performance analysis of the SL Intf, ML Int, SL Router and ML Router IPs respectively. Finally, conclusions are presented in section VIII.

II. The New generation of FPGAs for Space

There are two main devices that belong to the previous generation of space-qualified FPGAs. Both devices offer inbuilt high-speed transceivers that allow for the different SpFi IPs to be implemented.

On the one hand, there is the non-volatile configuration memory Microchip RTG4 manufactured in a low power 65 nm process [5], which is radiation-hardened by design. Thus, the big advantage of using the RTG4 is that Triple Module redundancy (TMR) has been integrated in its fabric transparently to the user. TMR is a method consisting of using triple module redundancy or triple voting to implement registers. Each register is implemented by three flip-flops that "vote" to determine the final output signal of the register function. Using TMR increases the number of resources used by an IP, affecting area and potentially also timing because of the additional logic inserted.

On the other hand, there is the SRAM-based (volatile) Xilinx XQRKU060 [6] manufactured on a faster and more compact 20 nm process. This FPGA offers roughly four times the resources of the RTG4 but does not offer the same degree of hardening against radiation.

A new generation of FPGAs specifically targeting space applications has recently been or is in the process of being released. This new generation aims at even higher performance applications, which makes them ideal targets for using very high-speed communications protocols such as SpFi.

A. Microchip PolarFire

The radiation-tolerant PolarFire (PF) RTPF500T FPGA is directly derived from its commercial counterpart, a non-volatile FPGA built on a 28 nm process [7]. PF uses low-power SONOS configuration switches that have been demonstrated to be robust at 100 krad of total dose and having an absence of configuration upsets under heavy-ion single event tests. Like the RTG4, PF provides 24 transceivers but with a higher maximum speed, each capable of running up to 10 Gbit/s. Unlike radiation-hardened devices, depending on the application the Single Event Upset (SEU) rate of the PF registers may not be good enough. In this case, the use of some form of TMR is advised.

B. Xilinx Versal

The Xilinx Versal XQRVC1902 [8] is the radiation-tolerant version of the commercial SRAM-based XCVC1902 FPGA. It is manufactured in a 7nm FinFET technology and provides a platform aiming at high performance applications, offering 44 GTY transceivers, each capable of running at more than 25 Gbit/s. Using TMR in Versal is also advised depending on the application and its requirement for register SEU sensibility, as the FPGA is not radiation-hardened.

C. NanoXplore BRAVE

The NanoXplore BRAVE FPGA family is a the European addition to the available options of space-qualified devices. There are several members of the BRAVE family, although only the NG-Large [9] (65nm FD-SOI SRAM) and NG-Ultra [10] (28 nm FD-SOI SRAM) include the inbuilt SerDes blocks required by SpFi. Both devices are radiation hardened by design, which removes the need for TMR.

III. SpaceFibre IP Cores General Features

The SpFi IP core family has been extensively tested in the different space FPGA families. IPs have been carefully designed to guarantee timing closure in all the temperature and voltage conditions required by the space devices, including EDAC in the memories and Single Event Transient (SET) filtering enabled—when available. These radiation mitigation techniques have an impact on the maximum speed of the designs and can potentially create problems to meet the targeted clock frequencies.

Effort has also been put to minimise the designer effort when adding the SpFi IP to a design. IPs are provided with a protocol agnostic data interface, so that no prior knowledge of the SpFi standard is required. Simple data interfaces based on standard 32-bit input and output FIFO interfaces are used. Specifically, they follow the AXI4-Stream (AXI4-S) protocol [11], which is a popular industry standard. This AXI4-S interface allows using independent user-defined read and write clocks, with clock synchronisation between user and SpFi IP clock domains managed by the IP. The AXI4-S width can be extended in 32-bit multiples in the ML Intf/Router IPs.

The IPs can be configured using generics. Different properties can be configured, e.g. transceiver interface, target technology for memory direct instantiation (for EDAC use), number and size of VCs, etc. Different high-speed transceiver interface options provide the set of signals to be directly connected to the selected transceiver. There are specific interfaces for the RTG4, PolarFire, 40-bit and 20-bit parallel, and Xilinx devices. Each of these takes into account whether 8B10B encoding/decoding, bit and symbol alignment, and clock correction can be done by the transceiver for better resource usage. Support for old FPGA technologies that require external transceivers is also provided through a dedicated TLK2711-SP (Wizardlink) [12] interface. A wrapper is supplied for each of the different transceiver interfaces for user convenience.

The QoS is independently and dynamically configurable for each VC, offering three mechanisms that work concurrently: scheduling, priority and bandwidth reservation. The FDIR mechanisms automatically recover from transient, persistent and permanent (when ML is used) errors on the SpFi link. A transient error takes less than 3 µsec to recover. It does not affect the user data rate thanks to the embedded buffering inside the IPs. Other protections against errors include data and broadcast babbling node protection. A lane is automatically disconnected when the BER is worse than 10^{-5} to prevent a potential protocol breakdown.

A management interface allows real time configuration of the IP control and status parameters, also including optional statistics and debug signals. Two different types of management interfaces can be selected: AXI4-Lite and APB bus. A signal bus is also available in the interface IPs. The AXI4-Lite and APB bus have independent clock with clock synchronisation managed inside the IP for convenience. Independent signals for each status and configuration field are useful when an FPGA design needs direct access to the IP. Accessing these fields over an AXI4-Lite/APB bus simplifies the interface for designs that use a CPU or want a centralised access point to several interfaces, for example. Power management options have been considered. For example, it is possible of start one end of the link in a low-power mode waiting for the other end to become active.

Two radiation testing campaigns have been carried out in collaboration with Microchip for the SpFi SL and ML Interfaces in the RTG4 [13, 14]. The information gathered during the test campaigns has allowed for assessing and further refining the robustness of the IPs under radiation and their associated RTG4 reference designs.

The IP Cores are also ready for ASIC implementation. For example, the SL Intf was used in the RC64 many-core DSP ASIC (12 SpFi interfaces) developed by Ramon.Space [15]. Other ASICs are currently under development implementing the ML Intf and the Router IPs.

Full TMR has been applied to the IPs in the PF to assess its impact in performance. As expected, the resulting synthesis takes ~2.8 times the number of initial registers (it is slightly below 3 because synchronisers are not TMR'ed). The number of LUTs is increased by a factor of ~2. One possibility to reduce the impact of TMR on the IP is to apply partial TMR. The idea is to only protect the most critical parts of the IP, in this case the control logic. This way, SEUs can induce sporadic data errors at the receiver, but the operation of the protocol itself is rugged against these events. This alternative is a compromise between full TMR and no TMR at all, and can be appropriate for certain applications which can tolerate a certain rate of data errors. This partial TMR is an ongoing development for the IPs here presented.

Finally, STAR-Dundee has adapted its SpFi IP Cores to be compatible with the BRAVE family. There is an ongoing activity to validate the operation of the IP inside the NG-Large FPGA. A successful SpFi link has been established, allowing the correct transmission of data between a STAR-Fire Mk3 unit and the NG-Large development board (Fig. 1). However, retry events were observed during the IP validation. The cause for these retries is probably related to the clock scheme adopted, which uses a fabric clock as the SerDes reference clock due to hardware limitations on the experiment set-up. A new set-up with an external SerDes reference clock is in the process of being tested. Nevertheless, it is worth highlighting that despite the retry events no data errors appeared. This is an example of the resilience of SpFi against errors on the link.

A. *Timing Performance*

Timing provided by the synthesis tools is not accurate, as the final timing depends on the routing and placement of the IP inside the FPGA fabric. Testing these IPs in different configurations on different development boards have confirmed that the maximum lane speeds can be achieved with the RTG4 (3.125 Gbit/s) even for congested designs. The rest of FPGAs achieve lane rates beyond 6.25 Gbit/s with plenty of margin, thus allowing to operate at these high speeds even when using TMR on the design. The figures shown in the tables of next sections all include transmit and receive FIFOs.

IV. SpaceFibre Single-Lane Interface IP Core

The resources required by the SL Intf IP are detailed in Table I for a different number of VCs. As the table shows, the IP offers a compact design only requiring a small percentage of area for implementing a SpFi interface, even when multiple VCs are used. Even with full-TMR, the impact in area usage of a SpFi link will be limited. Note that adding an additional VC to the design has also a limited impact on the overall resource usage.

The IP resource usage for both NG-Large and NG-Ultra has been obtained with the latest tool release—NXMap v22.1.0.1. Usage is the same for both devices because the fabric of the FPGAs is essentially the same, so no differences are expected between them. Due to the continuous evolution of NanoXplore tools, timing results continue to improve although they are still trailing those of Microchip and Xilinx devices. Final results will be presented once the IPs have been fully validated in BRAVE.

TABLE I. SpFi Single-Lane Interface Resource Usage

	RTG4			**XQRKU060** *		
	LUT	***DFF***	***LSRAM***	***LUT***	***DFF***	***RAMB36***
1 VC	3316 2.2%	2365 1.6%	4 1.9%	1823 0.5%	2346 0.4%	4 0.4%
2 VCs	3960 2.6%	2946 1.9%	6 2.9%	2162 0.7%	2969 0.4%	6 0.6%
4 VCs	5389 3.5%	4114 2.7%	10 4.8%	2960 0.9%	4214 0.6%	10 0.9%

	RTPF500T *			**XQRVC1902** *		
	LUT	***DFF***	***LSRAM***	***LUT***	***DFF***	***RAMB36***
1 VC	2796 0.6%	2226 0.5%	8 0.5%	1687 0.2%	2272 0.1%	2 0.2%
2 VCs	3400 0.7%	2801 0.6%	12 0.8%	1985 0.2%	2824 0.2%	3 0.3%
4 VCs	4653 1.0%	3972 0.8%	20 1.3%	2796 0.3%	3923 0.2%	5 0.5%

	NG-Large			**NG-Ultra**		
	LUT	***DFF***	***RAM***	***LUT***	***DFF***	***RAM***
1 VC	2703 2.0%	2496 1.9%	8 4.2%	2703 0.5%	2496 0.5%	8 1.2%
2 VCs	3275 2.4%	3068 2.4%	12 6.3%	3275 0.6%	3068 0.6%	12 1.8%
4 VCs	4350 3.2%	4220 3.3%	20 10.4%	4350 0.8%	4220 0.8%	20 3.0%

* TMR not included.

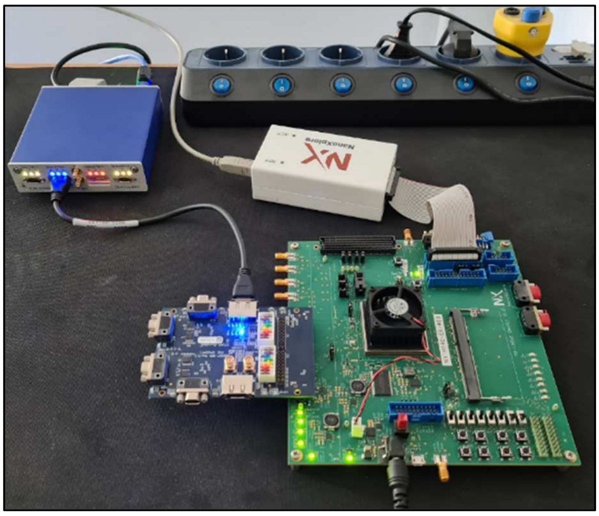


Fig. 1. BRAVE SpFi Interoperability Test with a STAR-Fire Mk3 unit.

V. SpaceFibre Multi-Lane Interface IP Core

A. Specific Features

Multi-lane is an optional capability of the SpFi link. The Multi- Lane layer coordinates the operation of multiple lanes acting as a single SpFi link, providing higher data throughput and redundancy. Each lane can be initialized and operated independently from each other.

The number of lanes is fully configurable, with any number of lanes supported (up to 16). Each lane can independently be selected as uni/bidirectional and hot/warm redundant. SL implementations must be bidirectional even if the end-user data flow is unidirectional, because of the feedback required by the protocol. However, in a Multi-Lane implementation only one bidirectional lane is enough for the interchange of protocol related information. Therefore, other lanes can be unidirectional to save power and mass in asymmetric data flows. The width of the AXI4-S interface of the VCs is configurable in multiples of the SpFi word size (32-bits). This allows supporting slower user clocks and still being able to send or receive data at the maximum speed over a single VC.

Hot redundant lanes allow the link to fully recover not only from transient errors (like the SL Intf), but also from persistent or permanent lane failures in less than 3 µs without user intervention and without any data loss. This 3 µs time is close to the round-trip delay of the lane. In case of lane failure in a link without redundant lanes, the link is automatically reconfigured to continue with the remaining working lanes, hence producing an automatic graceful degradation of the link bandwidth. The QoS mechanism ensures that the most important data is sent first, i.e. higher priority VCs or scheduled traffic are less affected. Warm redundant lanes save power with respect the always-on hot-redundant alternative, but they take around 20-40 µs to reach a working state. Bandwidth overprovision and dynamic power management are also possible. These capabilities are very useful for space applications where strict power constrains and a high level of reliability is required on the harsh space environment.

Fig. 2 shows the ML Interface IP being tested in a PF with 2 lanes coming out of a STAR-Dundee SpW/SpFi FMC daughterboard. Similarly, Fig. 3 also shows a ML testing on Versal with a set-up allowing for up to 8 lanes out of the FPGA by using 2 QSFP+ connectors on an FMC daughterboard.

B. Area Resources

Table II provides the FPGA resource usage for a combination of different number of lanes (2, 4 and 8) and VCs (1, 2 and 4). Individual lanes can operate up to 3.125 Gbps in the RTG4 and in excess of 6.25 Gbit/s in the rest of devices. This means aggregate rates with 8 lanes of up to 25 Gbit/s in the RTG4 and 50+ Gbit/s in the other devices. The user data rate (removing 8B10B and protocol overheads) that can be achieved in a full-duplex 8-lane scenario in each direction is 18.5 Gbit/s for the RTG4 and 37 Gbit/s for the rest of FPGAs. Multi-lane is a convenient way of multiplying the link bandwidth. It provides additional advantages, e.g. graceful degradation, unidirectional operation, redundancy, that are automatically managed by the link without a big increase in resources.

TABLE II. SpFi Multi-Lane Interface Resource Usage

	RTG4			**XQRKU060** *		
	LUT	***DFF***	***LSRAM***	***LUT***	***DFF***	***RAMB36***
2 Ln 1 VC	6870 4.5%	5166 3.4%	8 3.8%	3390 1.0%	4771 0.7%	8 0.7%
2 Ln 2 VCs	7792 5.1%	6166 4.1%	12 5.7%	3928 1.2%	5962 0.9%	12 1.1%
2 Ln 4 VCs	9492 6.3%	8087 5.3%	20 9.6%	4948 1.5%	7441 1.1%	20 1.9%
4 Ln 1 VC	12776 8.4%	9007 5.9%	16 7.7%	6020 1.8%	7942 1.2%	12 1.1%
4 Ln 2 VCs	13908 9.2%	10497 6.9%	24 11.5%	6744 2.0%	9259 1.4%	18 1.7%
4 Ln 4 VCs	15969 10.5%	13390 8.8%	40 19.1%	8100 2.4%	11792 1.8%	30 2.8%
8 Ln 1 VC	27203 17.9%	16739 11.0%	32 15.3%	12957 3.9%	14305 2.2%	20 1.9%
8 Ln 2 VCs	28565 18.8%	19197 12.6%	48 23.0%	13995 4.2%	16409 2.5%	30 2.8%
8 Ln 4 VCs	31076 20.5%	24014 15.8%	80 38.3%	15976 4.8%	20494 3.1%	50 4.6%

	RTPF500T *			**XQRVC1902** *		
	LUT	***DFF***	***LSRAM***	***LUT***	***DFF***	***RAMB36***
2 Ln 1 VC	4778 1.0%	4332 0.9%	12 0.8%	3195 0.4%	4611 0.3%	8 0.8%
2 Ln 2 VCs	5681 1.2%	5242 1.1%	18 1.2%	3699 0.4%	5444 0.3%	12 1.2%
2 Ln 4 VCs	7357 1.5%	6980 1.5%	30 2.0%	4629 0.5%	6995 0.4%	20 2.1%
4 Ln 1 VC	8619 1.8%	7381 1.5%	20 1.3%	5828 0.6%	7640 0.4%	12 1.2%
4 Ln 2 VCs	9724 2.0%	8681 1.8%	30 2.0%	6539 0.7%	8786 0.5%	18 1.9%
4 Ln 4 VCs	11733 2.4%	11189 2.3%	50 3.3%	7597 0.8%	10954 0.6%	30 3.1%
8 Ln 1 VC	19287 4.0%	13498 2.8%	36 2.4%	12703 1.4%	13712 0.8%	20 2.1%
8 Ln 2 VCs	20386 4.2%	15572 3.2%	54 3.6%	13410 1.5%	15676 0.9%	30 3.1%
8 Ln 4 VCs	23073 4.8%	19604 4.1%	90 5.9%	14876 1.7%	18865 1.0%	50 5.2%

	NG-Large **			**NG-Ultra** **		
	LUT	***DFF***	***RAM***	***LUT***	***DFF***	***RAM***
2 Ln 1 VC	5702 4.2%	5373 4.2%	16 8.3%	5702 1.1%	5373 1.1%	16 2.4%
2 Ln 4 VC	7878 5.7%	8410 6.5%	40 20.8%	7878 1.5%	8410 1.7%	40 6.0%
4 Ln 1 VC	10604 7.7%	9367 7.3%	32 16.7%	10604 2.0%	9367 1.9%	32 4.8%
4 Ln 4 VC	13254 9.7%	13926 10.8%	80 41.7%	13254 2.5%	13926 2.8%	80 11.9%
8 Ln 1 VC	22578 16.5%	17409 13.5%	64 33.3%	22578 4.2%	17409 3.4%	64 9.5%
8 Ln 4 VC	25793 17.3%	24975 19.4%	160 83.3%	25793 4.8%	24975 4.9%	160 23.8%

* TMR not included.

** Inferred values.

Regarding the BRAVE values indicated in Table II, for this IP and the Router IPs the resource usage has been inferred from the RTG4 values. Both BRAVE and RTG4 use LUT4 elements. It has been verified with the SL Intf IP that the resource increase ratio from 1 VC to 2 or 4 VCs is almost identical for BRAVE and RTG4. Hence, the usage ratio between RTG4 and BRAVE SL Intf IP has been used to infer the resources for the NG-Large/Ultra. Only results for 1 and 4 VCs have been included in the table for simplicity.

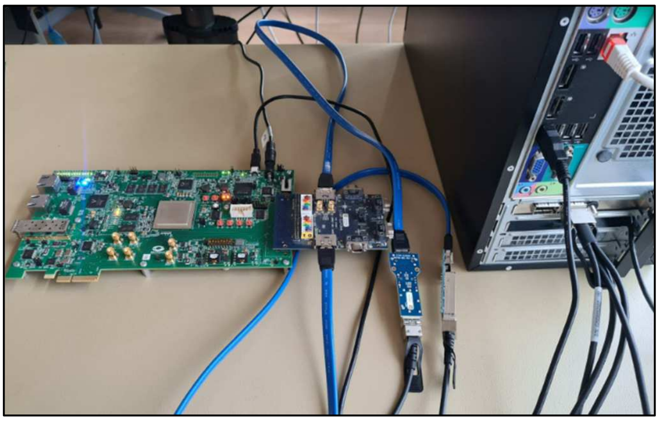


Fig. 2. SpFi ML on a PolarFire connected to a STAR-Ultra PCIe unit.

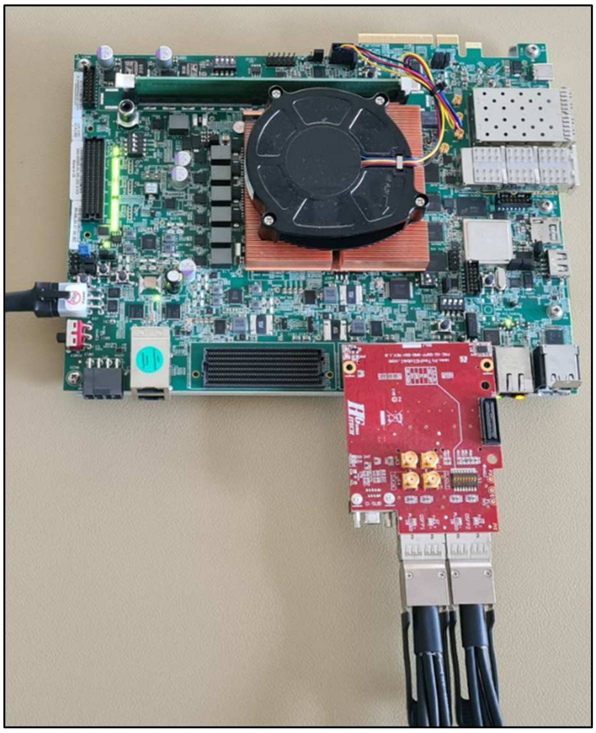


Fig. 3. SpFi Multi-Lane interface running on a Versal.

VI. SpaceFibre Single-Lane Router IP Core

The Router architecture is built around a non-blocking routing switch matrix with a configurable number of ports. Ports can be either SpFi, SpW or AXI4-S interfaces. Each port implements a configurable number of VCs. Each VC has an associated VN number. The switch matrix interconnects one or more VCs with the same VN number, with the limitation that each of these VCs must be in a different port. The output port is selected using path or logical addressing, indicated by the leading byte of each packet and the configuration of the internal routing table. Packets belonging to different VNs never interfere with one another and do not impact the throughput and latency within the routing switch matrix. On the other hand, when multiple packets in the same VN need to be transferred to the same output port, round-robin arbitration is performed packet by packet, like a SpW Router.

Fig. 4 shows a simplified Router. Note that SpW ports only have associated a single VC. The configuration port uses the RMAP protocol [16] to configure the routing table, VNs, Router ports, etc. Nevertheless, a dedicated AXI4-Lite interface can also be used to access the same configuration registers.

A. Specific Features

The STAR-Dundee SpFi SL Router IP is a scalable, fully configurable non-blocking router. The IP is very flexible, allowing to select the number of VCs, ports and target technology, among other options, using generics. The SpFi lane rates are also configurable. This Router implements path and logical addressing, VNs, time distribution and message broadcasting. In addition, it also fully supports the QoS and FDIR capabilities native to SpFi. The maximum number of VNs is 64, but each of these VNs is completely flexible: any VC of any port can be configured to any VN. VNs can be statically or dynamically configured. The VNs can be configured statically during FPGA programming using VHDL constants—allows using the Router IP without using any software host —, or they can be dynamically modified by the user using logic connected to the configuration port or the RMAP protocol, which can be accessed over one of the ports of the Router. The high flexibility of the SpFi Router IP Core ensures that different user needs can be accommodated with ease.

There are up to 256 broadcast channels with higher priority for time-critical broadcast messages. The Router offers a simple and efficient integration with SpW networks using SpW packet buffers and automatic SpW to SpFi broadcast translation. An internal timer tracks time being distributed over the network.

The Router IP presents a deterministic low latency switching. Round-robin packet arbitration can only occur within each VN. When arbitration is required, it only takes place when two or more VCs request to access to the same output port within the same VN. A timeout controls if the source or the sink stall in the middle of a packet, or when there is a babbling node. Upon timing-out, the router performs automatic packet spilling of the blocking packet.

B. Area Resources

Table III presents the resource usage for two different Router configurations which have been adopted as reference. The table shows that even a "large" Router of 8 SpFi ports with 4 VC each, plus SpW and AXI4-S ports, would fit inside an RTG4. The port count in the table includes two non-SpFi ports: one SpW port and one AXI4-S port with 2 VCs, that is 3 more VCs. So, for example, the 6 Port Router has 4 SpFi ports plus the SpW and AXI4-S ports. The SpFi SL Interface logic of the ports is included in the table figures, as well as the additional RMAP configuration port—one extra VC—and all the configuration logic (see Fig. 4). Hence, the total number of VCs of the Router is the total number of SpFi VCs plus 4.

Dividing the total number of VCs (12 and 36) of the two scenarios by the resource usage of the different FPGAs produce an interesting result. For all scenarios and devices, the average number of registers per VC is 2200±100. Regarding LUTs, their number per VC is 2600±100 for LUT4 (RTG4 and PF) and 1500±50 for LUT6 (KUS and Versal). This provides an easy method to calculate a rough estimation for a Router with a different number of ports and VCs.

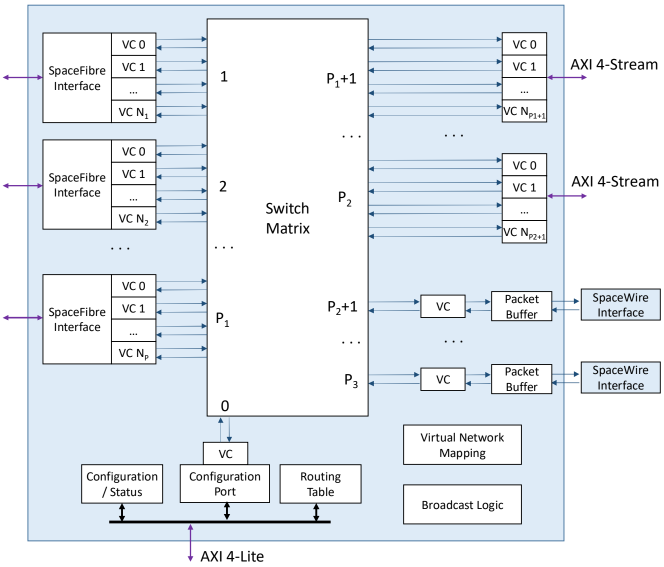


Fig. 4. SpaceFibre Router Block Diagram.

There are options to reduce the resource usage by tailoring the behaviour of the Router VNs. For some applications, provided that the network requirements are fixed, it is possible to limit the number of ports that can be accessed by a certain VN. This way VNs are pre-defined during the design phase and hardcoded into the Router design. This knowledge can be ported to the IP as a constraint, thus helping to reduce the area of the design.

VII. SpaceFibre Multi-Lane Router IP Core

The STAR-Dundee SpFi ML Router IP is directly derived from the SL Router. It provides the same functionality but with a configurable number of lanes on the SpFi ML ports. The main difference, apart from the ML interfaces, is that the internal data path width is increased accordingly—including the AXI4-S internal ports—so that the internal clock frequency of the Router does not scale up with the number of lanes. Thus, the internal clock frequency of the ML Router is the same of the SL Router. This multiplies the bandwidth of the Router at the expense of more resources but leaving timing largely unaffected. Note that the increase in congestion can have an impact on the final timing.

This IP has been used to build the primary element of the Hi-SIDE project, the STAR-Tiger, a 10 SpFi port ML Router with 4- and 2-lane ports [17]. In [18] there is an in-depth technical analysis of the ML Router architecture, operation and performance measurements, including latency (packet, switching, broadcast) or throughput depending on the packet size.

The scenarios analysed for the ML Router (Table IV) are identical to the SL Router, with the difference that all SpFi ports have either 2 or 4 lanes. The port count in the table also includes the SpW and AXI4-S ports. The values reported already include the SpFi ML InterfaceIP Cores used by each port and the additional configuration port (see Fig. 4).

TABLE III. SpFi Single-Lane Router Resource Usage

	RTG4			XQRKU060 *		
	LUT	*DFF*	*LSRAM*	*LUT*	*DFF*	*RAMB36*
6 Port 2 VCs	31782 20.9%	27393 18.0%	47 22.5%	17984 5.4%	28090 4.2%	48 4.4%
10 Port 4 VCs	98540 64.9%	76035 50.1%	127 60.8%	55917 16.9%	78051 11.8%	128 11.9%

	RTPF500T *			XQRVC1902 *		
	LUT	*DFF*	*LSRAM*	*LUT*	*DFF*	*RAMB36*
6 Port 2 VCs	29938 6.2%	26943 5.6%	93 6.1%	17098 1.9%	27652 1.5%	48 5.0%
10 Port 4 VCs	93526 19.4%	75905 15.8%	253 16.6%	53800 6.0%	75867 4.2%	128 13.2%

	NG-Large **			NG-Ultra **		
	LUT	*DFF*	*RAM*	*LUT*	*DFF*	*RAM*
6 Port 2 VCs	26379 19.2%	28489 22.1%	94 49.0%	26379 4.9%	28489 5.6%	94 14.0%
10 Port 4 VCs	81788 59.7%	79076 61.3%	254 132.3%	81788 15.2%	79076 15.6%	254 37.8%

SpFi Interface IP resources are included.

* TMR not included.

** Inferred values.

TABLE IV. SpFi Multi-Lane Router Resource Usage

	RTG4			XQRKU060 *		
	LUT	*DFF*	*LSRAM*	*LUT*	*DFF*	*RAMB36*
2L6P 2 VCs	48043 31.6%	44434 29.3%	59 28.2%	28579 8.6%	42829 6.5%	33.5 3.1%
2L10P 4 VCs	139644 92.0%	116463 76.7%	171 81.2%	82625 24.9%	109168 16.5%	101.5 9.4%
4L6P 2 VCs	77279 50.9%	69216 45.6%	117 56.0%	46808 14.1%	65607 9.9%	61.5 5.7%
4L10P 4 VCs	-	-	-	128600 38.8%	158420 23.9%	185.5 17.2%

	RTPF500T *			XQRVC1902 *		
	LUT	*DFF*	*LSRAM*	*LUT*	*DFF*	*RAMB36*
2L6P 2 VCs	47042 9.8%	44809 9.3%	67 4.4%	26492 2.9%	42844 2.4%	33.5 3.5%
2L10P 4 VCs	135690 28.2%	117563 24.4%	203 13.4%	77366 8.6%	109295 6.1%	101.5 10.5%
4L6P 2 VCs	76001 15.8%	69750 14.5%	123 8.1%	43106 4.8%	65608 3.6%	61.5 6.4%
4L10P 4 VCs	212500 44.2%	174216 36.2%	371 24.4%	120865 13.4%	158510 8.8%	185.5 19.2%

	NG-Large **			NG-Ultra **		
	LUT	*DFF*	*RAM*	*LUT*	*DFF*	*RAM*
2L6P 2 VCs	46278 33.8%	49729 38.5%	118 61.5%	46278 8.6%	49729 9.8%	118 17.6%
2L10P 4 VCs	130332 95.1%	128230 99.4%	342 178.1%	130332 24.3%	128230 25.4%	342 50.9%

SpFi Multi-Lane Interface IP resources are included.

* TMR not included.

** Inferred values.

The RTG4 does not allow for many SpFi ML ports as it has not enough resources for the design to fit. However, the rest of the FPGAs can implement large Routers using 4-lane SpFi ports with a total of 36 VCs (no TMR) while still having a considerable amount of free space for other applications if required. Comparing the resources for 2-lane and 4-lane ML Routers against the SL Router implementation shows that switching from single to 2-lane SpFi ports requires roughly ~50% more resources. Moving to a 4-lane version instead increases resources demand by ~150%. Note that in a 2-lane version the bandwidth of the Router is doubled and that in the 4-lane case the bandwidth is multiplied by 4.

Regarding the ratio of resources/VC, for the 2-lane a rough order of magnitude is ~3500 registers/VC and ~3900 LUT4/VC or ~2200 LUT6/VC. For the 4-lane case, the ratios are ~4500-5500 registers/VC (the more VCs the lower the ratio) and ~6000 LUT4/VC or ~3500 LUT6/VC.

Finally, note that the last Router configuration scenario (4L10P) requires 32 lanes. The PF FPGA, despite having ample margin to implement such configuration only offers 24 SerDes lanes, so it is not possible to get all these 4-lane SpFi ports out of the FPGA. The values for this configuration have not been added to the RTG4 table because it exceeds available resources.

VIII. Conclusions

The STAR-Dundee SpaceFibre Single-Lane Interface, Multi-Lane Interface, Single-Lane Router and Multi-Lane Router IP Cores have been designed to be easy to implement in radiation-tolerant FPGAs. In this article we have detailed the performance and capabilities of the different IP Cores, and discussed the resources required depending on several parameters, namely the number of VCs, lanes and ports.

A simple SpFi Single-Lane Interface (1 VC) can be integrated in radiation-hardened FPGAs by using only a 2% of an RTG4 and less than 0.5% in the other FPGAs analysed. This offers a simple way of having a high-speed and resilient communication channel with an FPGA.

The multi-lane capability increases the data throughput of SpFi and the addition of multiple lanes provide hot and warm redundancy, or graceful degradation of the link bandwidth when no redundant lanes are available. Therefore, if more bandwidth or additional robustness is required out of a SpFi link, the Multi-Lane Interface IP is a convenient choice, keeping resource usage at a mere 4% for the RTG4 and 1% or less for the other devices. Finally, the SpFi Routers (SL and ML) can also be integrated in space-qualified FPGAs even when many ports are required.

All the STAR-Dundee IP Cores have been verified in simulation and subsequently validated in hardware prototypes. Both commercial and the main radiation-hardened FPGAs have been used for these validation activities, ensuring full compatibility, and defining an easy adoption path for this technology. IPs come with specific reference designs for each FPGA, and these can directly be implemented in the FPGA to assist the end-user and allow an easy adoption. A comprehensive end user test bench for ModelSim/Questa simulators is also provided, which can be used as a reference for test integration.

These IPs provide the all the necessary building blocks for creating next generation of onboard networks. This has been demonstrated in the Hi-SIDE project, a European Union project involving several European aerospace organisations that have developed satellite data-chain technologies for future Earth observation and telecommunication systems [17]. The different elements of the data chain are interconnected via a SpFi network. SpFi is currently being implemented in FPGA and ASIC designs by different missions and products all over the world.

References

[1] ECSS Standard ECSS-E-ST-50-11C, "SpaceFibre – Very high-speed serial link", European Cooperation for Space Data Standardization, 15th May 2019, available from http://www.ecss.nl.

[2] ECSS Standard ECSS-E-ST-50-12C, "SpaceWire, Links, Nodes, Routers and Networks", Issue 1, European Cooperation for Space Data Standardization, July 2008, available from http://www.ecss.nl.

[3] M. Farras Casas, S. Parkes, A. Ferrer Florit and A. Gonzalez Villafranca, "Testing SpaceFibre in Orbit: the OPS-SAT and NORBY Technology Demonstrators", 2022 International SpaceWire Conference.

[4] A. Ferrer Florit, A. Gonzalez Villafranca, S. Parkes and C. McClements, "SpaceFibre Interface and Routing Switch IP Cores", 2018 International SpaceWire Conference, available from https://www.star-dundee.com/wp-content/star_uploads/conference_papers/spacefibre/2018_SPW_SpaceFibre_Interface_Routing_Switch_IP_Cores.pdf (last visited 11th September 2022).

[5] Microsemi, "RTG4 FPGAs Technical Brief", rev B, Microsemi, October 2021, available from https://www.microsemi.com/document-portal/doc_download/134430-pb0051-rtg4-fpgas-product-brief (last visited 11th September 2022).

[6] Xilinx, "Radiation Tolerant Kintex UltraScale XQRKU060 FPGA Data Sheet DS882", v1.3, Xilinx, 11th April 2022, available from https://docs.xilinx.com/v/u/en-US/ds882-xqr-kintex-ultrascale (last visited 11th September 2022).

[7] Microchip, "PO0145 Product Overview Radiation-Tolerant PolarFire FPGA", Microchip, 2019, available from https://www.microsemi.com/document-portal/doc_download/1244478-rt-polarfire-fpga-product-overview (last visited 11th September 2022).

[8] Xilinx, "XQR Versal ACAP Product Table", Xilinx, 2021-2022, available from https://china.xilinx.com/content/dam/xilinx/support/documents/selection-guides/xqr-versal-product-selection-guide.pdf (last visited 11th September 2022).

[9] NanoXplore, "NG-Large Space NX1H140ATSP Datasheet", ver 1.0.3, NanoXplore, May 2022, available from https://files.nanoxplore.com/f/db44862a31ff4521bb4a/?dl=1 (last visited 11th September 2022).

[10] NanoXplore, "NG-Ultra NX2H540TSC Datasheet", ver 2.1, NanoXplore, June 2022, available from https://files.nanoxplore.com/f/5785df61b4f545a08ae8/?dl=1 (last visited 11th September 2022).

[11] ARM, "AMBA AXI Protocol Version 2.0 Specification", ARM, Ed., 2010.

[12] Texas Instruments Inc., "1.6-Gpbs to 2.5-Gbps Class V Transceiver", Texas Instruments Inc., 2018, available from http://www.ti.com/product/tlk2711-sp (last visited 11th September 2022).

[13] A. Gonzalez Villafranca, A. Ferrer Florit, M. Farras Casas, S. Parkes and C. McClements, "SpaceFibre for FPGA: IPs and Radiation Test Results", SEE-MAPLD 2020.

[14] A. Gonzalez Villafranca, A. Ferrer Florit, M. Farras Casas and N. Rezzak, "Mitigation and Recovery of Single Event Effects in RTG4 FPGA Transceiver Links Using SpaceFibre", RADECS 2022.

[15] R. Ginosar, P. Aviely, T. Israeli and H. Meirov, "RC64: High Performance Rad-Hard Manycore", IEEE Aerospace Conference, Big Sky, Montana, 2016.

[16] ECSS Standard ECSS-E-ST-50-52C, "SpaceWire – Remote Memory Access Protocol", European Cooperation for Space Data Standardization, 5th February 2010, available from http://www.ecss.nl.

[17] S. Parkes et al., "SpaceFibre Payload Data-Handling Network", 2022 International SpaceWire Conference.

[18] A. Ferrer Florit, A. Gonzalez Villafranca, M. Farras Casas and S. Parkes, "SpaceFibre Multi-Lane Routing Switch IP", 2022 International SpaceWire Conference.

SpaceFibre Multi-Lane Routing Switch IP

Albert Ferrer Florit
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
albert.ferrer@star-dundee.com

Alberto Gonzalez Villafranca
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
alberto.gonzalez@star-dundee.com

Marti Farras Casas
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
marti.farras@star-dundee.com

Steve Parkes
STAR-Dundee
Dundee, Scotland, UK
steve.parkes@star-dundee.com

***Abstract*— SpaceFibre (ECSS-E-ST-50-11C) is a very high-performance, high-reliability and high-availability network technology specifically designed to meet the needs of space applications. It provides point-to-point and networked interconnections at Gigabit rates with Quality of Service (QoS) and Fault Detection, Isolation and Recovery (FDIR). SpaceFibre has been designed as a replacement of SpaceWire (ECSS-E-ST-50-12C) – it is backwards compatible with SpaceWire at the packet level – for next-generation space missions where very high throughput is required. SpaceFibre provides up to 6.25 Gbit/s per lane, with multi-lane allowing to reach up to 16 times the speed of a single lane.**

In this work we present the SpaceFibre Multi-Lane Routing Switch IP Core developed by STAR-Dundee and its subsidiary STAR-Barcelona. This IP provides a highly flexible router comprising a number of ports and a fully configurable, non-blocking, high performance, routing switch matrix. The internal ports use AXI4-Stream protocol, and the external ports can implement SpaceFibre or SpaceWire interfaces. The SpaceWire ports include additional bridging logic for efficient interconnection between SpaceWire and SpaceFibre equipment. The core logic of the IP is technology independent but has been optimised to be easily implemented in radiation tolerant FPGAs.

The routing switch is fully compliant with all layers of the SpaceFibre standard, supporting up to 64 virtual networks and 256 broadcast channels. Among other features, it implements network time synchronisation, packet time-outs, and automatic translation between SpaceFibre broadcast messages and SpaceWire broadcast codes (SpaceWire Time-Codes or Interrupts). With up to 8 lanes per SpaceFibre interface, raw link rates of 50Gbps per port can be achieved.

The multi-lane routing Switch Ip Core is implemented in the STAR-Tiger Routing Switch of the Hi-SIDE project.

Keywords—SpaceFibre, Routing Switch, IP Core, FPGA, Radiation Tolerant, RTG4.

I. Introduction

SpaceFibre (SpFi) [1] is the next generation of SpaceWire (SpW) [2] network technology for use onboard spacecraft. It supports high data-rate payloads, provides robust, long-distance communications for launcher applications, and supports avionics applications with deterministic delivery capability. SpaceFibre provides in-built Quality of Service (QoS) and Fault Detection, Isolation and Recovery (FDIR) capabilities and runs over electrical or fibre-optic cables.

A. SpaceFibre Lanes

New generation payloads, such as SAR and multi-spectral imaging instruments, require the use of multiple parallel high-speed links to fulfil the increasing bandwidth requirements. To accommodate these needs, SpFi supports multi-lane operation, thus allowing data to be sent over several individual physical lanes to enhance throughput and robustness. The multi-lane operation allows much higher data rates through lane aggregation, supporting any number of lanes (up to 16) and unidirectional operation. This effectively multiplies the throughput of the interface by combining several lanes into a single link. Furthermore, when a lane fails the multi-lane mechanism supports hot redundancy and graceful degradation by automatically spreading traffic over the remaining working lanes.

B. SpaceFibre Link

A SpFi link is made up of one or more lanes. In a multi-lane link, some of the lanes can be unidirectional provided that at least one lane is bi-directional. The SpFi link provides QoS and error recovery. SpFi links carry traffic (application information) through one or more virtual channels (VCs). There is a maximum of 32 VCs on a link. Traffic entering VC N comes out of VC N at the other end of the link.

Each VC is provided with a QoS which has three components: bandwidth reservation, priority and scheduling. Bandwidth reservation, reserves a portion of the link bandwidth for the VC. Priority assigns a priority-level to the VC so that higher priority VCs are able to send before lower priority ones. Scheduling divides time into 64 sequential time-slots and specifies in which of those time-slots a VC is permitted to send information. These three different QoS components are not alternatives, they work together.

C. SpaceFibre Network

SpFi carries SpW packets over VCs and provides a broadcast feature similar to SpW time-codes but offering much more capability.

The information to be sent is packaged in the same packet format as SpW. SpFi also uses the same routing concepts as SpW including both path and logical addressing. SpFi broadcasts are short messages that are expected to be received by all nodes of the network with minimum latency and jitter. Each broadcast source must use a different broadcast channel.

Fig. 1 shows an example onboard network using SpFi. The control processor is used to configure and control all on-board data-handling equipment. It therefore needs a connection to every instrument and to the mass memory. This could be provided using a separate command and control network, but this would result in additional mass and power consumption. The addition of a SpFi Routing Switch connecting to all of the on-board equipment allows the control processor to send commands and receive information from all of the on-board data handling units.

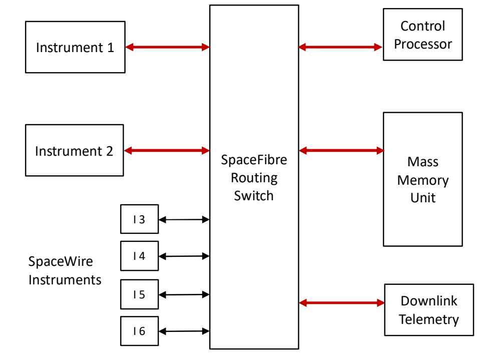


Fig. 1. Onboard network example using SpFi.

Each SpFi link implements multiple VCs, each one with specific QoS parameters. For example, the bandwidth allocation parameter of each VC can be set to the expected bandwidth of each instrument, so they do not interfere with one another or with the control network.

D. *SpaceFibre Virtual Networks*

The SpFi Network layer defines the concept of Virtual Network (VN). VNs are built from the interconnection between VCs of different ports. These VNs enable the creation of highly flexible SpFi routing switches comprising a number of SpFi interfaces and a fully configurable, non-blocking, high performance, routing switch. This routing switch can theoretically support an arbitrary number of VNs, each effectively behaving like independent SpW networks capable of working at multi-Gbps rates.

Fig. 2 shows a simple example of how the control network (blue path) and two instrument data flows (green and yellow paths) can be assigned to VNs.

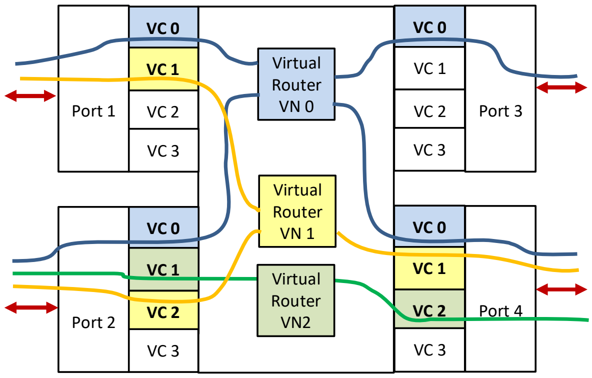


Fig. 2. Example of virtual networks within a SpFi Routing Switch

The traffic running over each VN is constrained by the SpFi QoS mechanism of each of its VCs. Traffic remains within its allocated bandwidth and follows the priority and schedule allocated to it. Data within a VN cannot flow to another VN.

A VN is also able to opportunistically use more bandwidth than it has been allocated, when no other VN has traffic to send over the links of the SpFi network that the particular VN wants to use.

As far as the addressing of packets and their routing across the network is concerned, SpFi operates in the same way as SpW. This has the substantial advantage that existing application software or SpW equipment can be used with a SpFi network by simply tying a SpW link interface to a SpFi VC interface. The application does not need to know that it is running over SpFi, but gains all the QoS and FDIR advantages of SpFi. This makes the integration of existing SpW equipment both simple and advantageous [3].

E. *SpaceFibre Multi-lane Routing Switch IP Core*

The STAR-Dundee SpFi Multi-Lane Routing Switch IP is directly derived from the STAR-Dundee SpFi single-lane Routing Switch IP [4]. The multi-lane version provides the same functionality but with a configurable number of lanes on the SpFi ports. The main difference, apart from the multi-lane capable interfaces, is that the internal data path width is increased accordingly—including the AXI4-Stream internal ports—so that the internal clock frequency of the Routing Switch does not scale up with the number of lanes. Thus, the internal clock frequency of the multi-lane version is the same of the single-lane. This multiplies the bandwidth of the Routing Switch at the expense of more resources, but leaving timing largely unaffected.

This paper presents the STAR-Dundee SpFi Multi-Lane Routing Switch IP Core (the Routing Switch). Its architecture and features are described in Section II and Section III respectively. Section IV and V presents the synthesis and performance results. An example of a hardware implementation is described in section VI. Finally, conclusions are presented in section VII.

II. ROUTING SWITCH ARCHITECTURE

The Routing Switch architecture is built around a non-blocking routing switch matrix with a number of ports, which can implement a SpW, SpFi or AXI4-Stream interfaces. Each port has a number of VCs, each one comprising an input and an output VC buffer. Each VC has an associated VN number. The switch matrix interconnects one or more VCs with the same VN number, but each of these VCs must be located in a different port. The output port is selected using path or logical addressing, indicated by the leading byte of each packet and the configuration of the internal routing table.

Packets belonging to different VNs never interfere with one another and do not impact the allocated throughput and latency within the routing switch matrix. On the other hand, when multiple packets in the same VN need to be transferred from different ports to the same output port, packet-by-packet, round-robin arbitration is performed, similarly to a SpW router.

Fig. 3 shows a simplified Routing Switch architecture with a configurable number of ports. The SpW ports only have associated a single VC but have additional buffering resources. The configuration port implements the RMAP protocol [5] to configure the Routing Table, the VNs, and the SpFi and SpW interfaces.

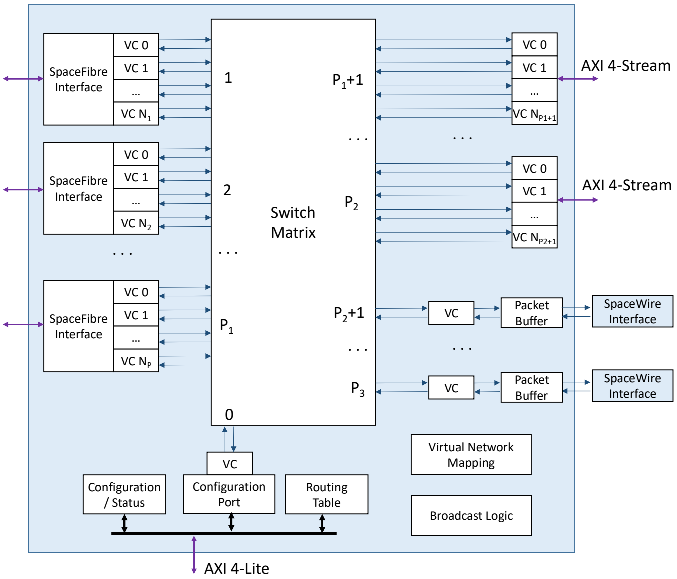


Fig. 3. SpaceFibre Routing Switch block diagram

The Switch Matrix allows to interconnect every VC from one port to any other VC from another port. The specific VN configuration will specify which connections are allowed in a particular application. The non-blocking switching logic allows to reach full bandwidth for each possible connection, independently of other simultaneous connections being used.

When a packet arrives at an input port from a particular VC, the output port is determined based on the packet address and the Routing Table, which translates logical address numbers to output port numbers. Then, the VC within this output port is determined by the VN Mapping block using the VN configuration information.

If an output VC has no space or it is busy transferring data from another packet coming from another input port within the same VN, the packet is stalled until the resource is freed. A round robin scheme ensures fair arbitration for packets from the same VN requesting the same output port at the same time.

Finally, the Broadcast Logic block handles how the broadcast messages received are sent across all allowed output ports, implementing the mechanism defined in the standard to avoid network level broadcast storms in the presence of switching loops.

III. Routing Switch Features

The Routing Switch has the following main features:

- Technology independent (FPGA or ASIC) but optimised for radiation-hardened FPGAs.
- Configurable number of SpFi, SpW and internal AXI4-Stream ports.
- Configurable SpFi lane rate, number of lanes, and number of VCs per port.
- Configurable target technology (RTG4, PolarFire, Xilinx Kintex/Ultrascale/Versal, generic) for memory blocks and SerDes interface.
- Up to 64 VNs that can be statically or dynamically configured.
- Configuration registers can be accessed via a configuration port using RMAP or using a dedicated AXI4-Lite interface.
- High performance, full non-blocking switch matrix with deterministic switching latency. VNs do not share any switching resources.
- Round-robin arbitration with watchdog timeout for packets in the same VN requesting the same output port.
- SpW/SpFi network capabilities such as path and logical addressing with a routing table.
- Up to 256 broadcast channels with higher priority for time-critical broadcast messages.
- Simple and efficient integration with SpW networks using SpW packet buffers and automatic SpW to SpFi broadcast translation.
- Internal timer tracks time being distributed over the network.

A. *Configuration*

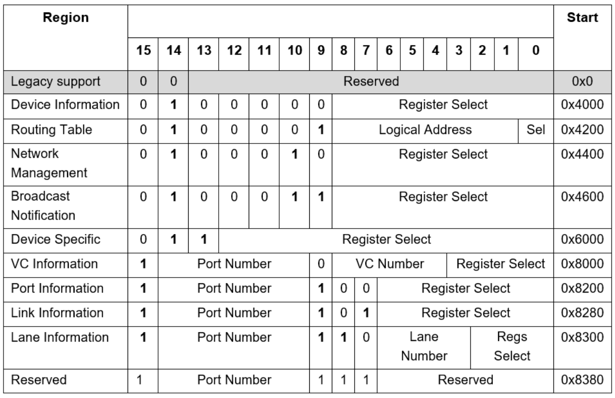
The main capabilities of the Routing Switch can be configured statically before the IP is synthetised. The most important are the target technology, number and type of ports, lane rate, lanes and VCs per port, the default value of the routing table, and the VN setup. This allows to use the Routing Switch with the default configuration, without using any software host, and to optimise router resources required for a specific application.

Each VN is configured by specifying the VC used for each port of the router in which the VN is used. Table I shows the VNs configuration table for the simple case shown in Fig. 2.

TABLE I. Virtual Network Configuration Example

VN number	**Virtual channel number**			
	Port 1	Port 2	Port 3	Port 4
0	0	0	0	0
1	1	2	-	1
2	-	1	-	2

Once the Routing Switch has been implemented it can be configured after reset by accessing to the router registers using the AXI4-Lite interface or the configuration port zero and the RMAP protocol. The Routing Switch memory space uses 16-bits address with either bit 15 or bit 16 set, to support network discovery of legacy devices such as the SpW 10-X Router ASIC (AT7910E) [6]. Fig. 4 shows the memory map regions of the Routing Switch.



Region																	Start
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Legacy support	0	0	Reserved														0x0
Device Information	0	**1**	0	0	0	0	0	Register Select									0x4000
Routing Table	0	**1**	0	0	0	0	**1**	Logical Address								Sel	0x4200
Network Management	0	**1**	0	0	0	**1**	0	Register Select									0x4400
Broadcast Notification	0	**1**	0	0	0	**1**	**1**	Register Select									0x4600
Device Specific	0	**1**	**1**	Register Select													0x6000
VC Information	**1**	Port Number					0	VC Number				Register Select					0x8000
Port Information	**1**	Port Number					**1**	0	0	Register Select							0x8200
Link Information	**1**	Port Number					**1**	0	**1**	Register Select							0x8280
Lane Information	**1**	Port Number					**1**	**1**	0	Lane Number				Regs Select			0x8300
Reserved	1	Port Number					1	1	1	Reserved							0x8380

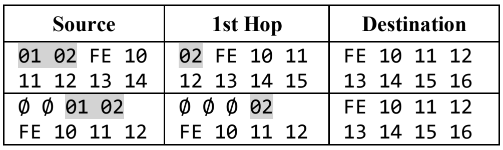
Fig. 4. Memory map regions of the Routing Switch

B. *Packet Addressing*

The routing control logic interprets the first byte of each received packet as the packet address. If the leading byte of a packet has a value between 0 and 31, path addressing is used and the packet will be forwarded to the corresponding port number. Otherwise, logical addressing is used, and the output port is determined by the routing table. Therefore, this works the same as a SpW router.

However, SpFi links also support leading FILL characters in a SpFi packet. When leading FILLs are present, the first packet byte is then replaced by a FILL character in case it is a path address value. If leading FILLs are not present, the first packet byte is removed when it is a path address value. This behaviour ensures packet cargo is 32-bit aligned when it arrives at the destination when the packet is transferred across SpW and SpFi networks using path address bytes, as SpW does not support sending FILL characters.

Fig. 5. shows an example for these two scenarios that the router supports. Note that the number of path address bytes matches the number of network hops, the FILL character is represented by symbol Ø and that the SpFi standard specifies that the router should remove 4 consecutive leading FILL characters.



Source	1st Hop	Destination
01 02 FE 10 11 12 13 14	02 FE 10 11 12 13 14 15	FE 10 11 12 13 14 15 16
Ø Ø 01 02 FE 10 11 12	Ø Ø Ø 02 FE 10 11 12	FE 10 11 12 13 14 15 16

Fig. 5. Examples of path address processing.

C. *SpaceWire Packet Buffers*

Each SpW port has a packet buffer on its receive side and a FIFO buffer on its transmit side. The Packet Buffer buffers packets arriving over the SpW port. It only forwards full packets to the SpFi VN. The packets arrive at the buffer at SpW speeds and leave at SpFi speeds, so that the VC sending the SpW packet is not held up by the slower SpW interface. If the incoming packet is larger than the size of the Packet Buffer there is a configurable option to spill the remaining of the packet. The FIFOs on the transmit side of the SpW interface simply buffer the traffic arriving over SpFi. The size of the buffers and FIFOs is configurable.

D. *Watchdog Timeout Mechanism*

SpFi VNs decouple the traffic flowing in one VN from the traffic in another VN. Therefore, if a packet becomes blocked in a VN it will not affect packets in another VN. Any congestion in a VN will not affect another VN.

However, within the same VN, package blocking can still occur, in the same way that there can be packet blocking and congestion in a SpW network. There are three main causes:

a) Source stalls and stops transmitting bytes of a SpW packet while the packet is being routed.

b) Destination stalls and stops receiving bytes of a SpW packet while the packet is being routed.

c) A package is blocked due to another packet being blocked. This can only occur if both use the same VC at one of the links of the path to their destination.

The Routing Switch implements a watchdog timer to prevent indefinitely packet blocking. When the packets transfer stops the watchdog timer is started. When the maximum time elapses, the packet is spilled.

E. *Broadcast messages*

The Routing Switch forwards any broadcast message type but there are some broadcast types that are processed in a specific manner:

- *Time*: Used to synchronise the local time with the network time. The CCSDS value of the time broadcast message is validated by comparing the value of two consecutive time broadcasts received. The time difference is compared with the difference in arrival time using the local clock. If the difference is very small the broadcast time value is accepted. The local time is then updated with this new validated value.

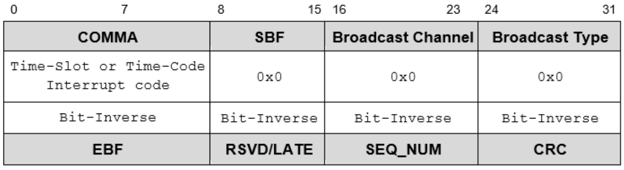
- *Time-Slot*: Used to set the device time-slot and synchronise time-slots across the network for SpFi network scheduling.

- *SpW Time-Code*: Contains a SpW broadcast code of type Time-Code. The Routing Switch generates this broadcast when a Time-Code is received in a SpW port. The broadcast generated is sent to all ports except the SpW port that received the Time-Code. Likewise, the Routing Switch distributes a SpW Time-Code to all the SpW ports when this broadcast is received from a SpFi port.

- *SpW Interrupt*: Contains a SpW broadcast code of type Distributed Interrupt. The Routing Switch generates this broadcast when a Distributed Interrupt is received in a SpW port. The broadcast generated is sent to all ports expect from the SpW port that received the Distributed Interrupt. Likewise, the Routing Switch sends a SpW Distributed Interrupt to all the SpW ports when this broadcast is received from a SpFi port.

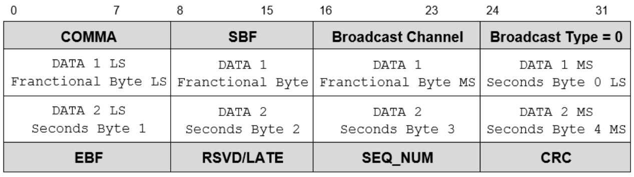
The broadcast type value of each of these broadcast types can be configured, with lower broadcast type values being forwarded with higher priority. Fig. 6 shows the format of the last three broadcast message types. The second 32-bit data word is the bit-inverse value of the first 32-bit data word. The least significant bits hold the actual value. SpFi Broadcasts messages generated by the router from SpW broadcast codes use the Broadcast channel specified in register "Router BC Channel". The value in this register must be initialised to enable the forwarding of SpW Time-Codes and Interrupts to SpFi broadcast messages.

Fig. 7 shows the Time broadcast message format which holds a CCSDS time information value.



0 7	8 15	16 23	24 31
COMMA	SBF	Broadcast Channel	Broadcast Type
Time-Slot or Time-Code Interrupt code	0x0	0x0	0x0
Bit-Inverse	Bit-Inverse	Bit-Inverse	Bit-Inverse
EBF	RSVD/LATE	SEQ_NUM	CRC

Fig. 6. Broadcast frame carrying a SpW broadcast code or SpFi time-slot.



0 7	8 15	16 23	24 31
COMMA	SBF	Broadcast Channel	Broadcast Type = 0
DATA 1 LS Franctional Byte LS	DATA 1 Fractional Byte	DATA 1 Franctional Byte MS	DATA 1 MS Seconds Byte 0 LS
DATA 2 LS Seconds Byte 1	DATA 2 Seconds Byte 2	DATA 2 Seconds Byte 3	DATA 2 MS Seconds Byte 4 MS
EBF	RSVD/LATE	SEQ_NUM	CRC

Fig. 7. Broadcast frame carrying a CCSDS time information.

IV. Synthesis Results

The Routing Switch has been designed to achieve timing closure at the highest data rates supported by the transceivers available in existing radiation-tolerant technologies. The IP supports lane rates of 3.125 Gbps in RTG4 and 6.25 Gbps in PolarFire FPGAs. In UltraScale and Versal Xilinx devices, faster speeds are possible.

The Routing Switch IP has also been designed to scale well in both timing and area metrics when the number of lanes, ports and VCs are increased. This means that the same maximum lane rates can be achieved independently of these parameters. Regarding area, Fig. 8 and Fig. 9 show that area resources increase linearly with these parameters. Note that this same slope is common to the number of lanes and VCs per port. A different slope—slightly above 1—is associated with the number of ports.

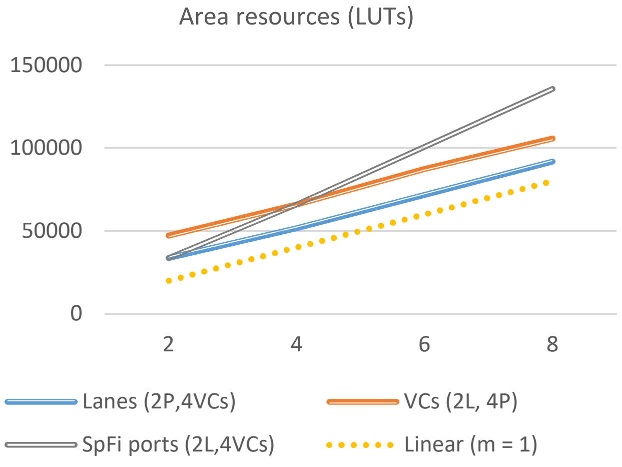


Fig. 8. Linear dependency of LUTs according to different parameters in a PolarFire implementation.

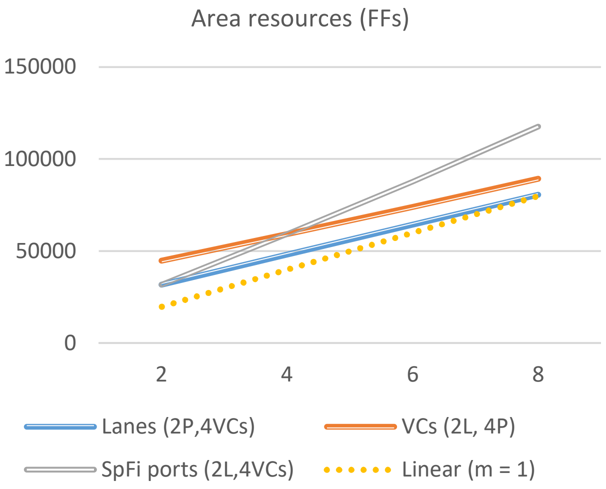


Fig. 9. Linear dependency of Flip-Flops according to different parameters in a PolarFire implementation.

Table II presents the resource usage after synthesis for different value of number of lanes, ports and VCs. The port count in the table includes two non-SpFi ports: one SpW port and one AXI4-Stream internal port with 2 VCs. The SpFi Multi-Lane Interface logic of the ports is included in the logic count, as well as the additional RMAP configuration port (see Fig. 3).

TABLE II. SpFi Multi-Lane Routing Switch Resource Usage

	RTG4			XQRKU060 *		
	LUT	*DFF*	*LSRAM*	*LUT*	*DFF*	*RAMB36*
2L6P **2 VCs**	48043 31.6%	44434 29.3%	59 28.2%	28579 8.6%	42829 6.5%	33.5 3.1%
2L10P **4 VCs**	139644 92.0%	116463 76.7%	171 81.2%	82625 24.9%	109168 16.5%	101.5 9.4%
4L6P **2 VCs**	77279 50.9%	69216 45.6%	117 56.0%	46808 14.1%	65607 9.9%	61.5 5.7%
4L10P **4 VCs**	-	-	-	128600 38.8%	158420 23.9%	185.5 17.2%

	RTPF500T *			XQRVC1902 *		
	LUT	*DFF*	*LSRAM*	*LUT*	*DFF*	*RAMB36*
2L6P **2 VCs**	47042 9.8%	44809 9.3%	67 4.4%	26492 2.9%	42844 2.4%	33.5 3.5%
2L10P **4 VCs**	135690 28.2%	117563 24.4%	203 13.4%	77366 8.6%	109295 6.1%	101.5 10.5%
4L6P **2 VCs**	76001 15.8%	69750 14.5%	123 8.1%	43106 4.8%	65608 3.6%	61.5 6.4%
4L10P **4 VCs**	212500 44.2%	174216 36.2%	371 24.4%	120865 13.4%	158510 8.8%	185.5 19.2%

	NG-Large **			NG-Ultra **		
	LUT	*DFF*	*RAM*	*LUT*	*DFF*	*RAM*
2L6P **2 VCs**	46278 33.8%	49729 38.5%	118 61.5%	46278 8.6%	49729 9.8%	118 17.6%
2L10P **4 VCs**	130332 95.1%	128230 99.4%	342 178.1%	130332 24.3%	128230 25.4%	342 50.9%

SpFi Multi-Lane Interface IP resources are included.

* TMR not included.

** Inferred values.

V. Performance Results

The Routing Switch architecture supports the full bandwidth of the SpFi interfaces without any dependency on the number of simultaneous data flows within the router. There is only potential blocking when two input ports request the same output port within the same VN. Therefore, the two main performance figures, the packet latency and the data packet throughput, can be easily obtained.

Table III shows the packet latency measured in clock cycles since the start of a packet entering an input port until the start of the packet appearing at the output port. The switching latency only takes into account the time a packet stalls until the output port is determined. This time determines the packet data throughput. The use of logical addressing increases both latency values due to the need to access the routing table.

TABLE III. Routing Switch latency

	Path addressing	**Logical Addressing**
Packet Latency	34 clock cycles	38 clock cycles
Switching Latency	22 clock cycles	25 clock cycles
Broadcast Latency	10 Clock cycles	

The Routing Switch has been optimised for timing and to scale well when the number of lanes, ports, and VCs is increased. The trade-off is more pipelining, which increases latency. For example, at 6.25 Gbit/s lane rate and with a core clock of 156.25 MHz, the packet latency is around 243 ns. However, for fast FPGAs or ASICs, the core clock of the Routing Switch can be much faster than the one used by the SpFi interfaces so the latency can be reduced in exchange for more power utilisation.

The broadcast latency of low priority broadcast types can be higher than the one shown if there are other higher priority broadcast pending to be sent. This increases the jitter of this lower priority broadcast but the broadcast message is then modified by the router with the delayed status flag set, so the destination knows that the broadcast has been delayed by the router.

Fig. 10 shows the data throughput depending on the packet size.

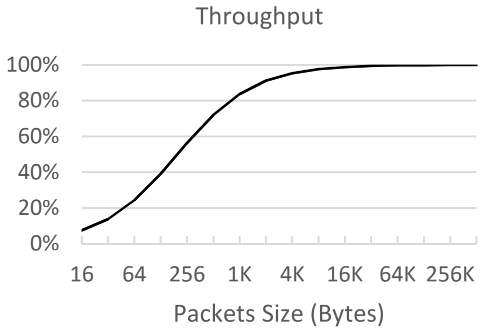


Fig. 10. Throughput depending on the size of the packets.

VI. Hardware Implementation

The SpFi Multi-Lane Routing Switch IP has been used to build the primary element of the Hi-SIDE project, the STAR-Tiger, a 10-port SpFi Multi-Lane Routing Switch with 4-lane and 2-lane ports [7].

The Hi-SIDE project is a European Union project carried out by several leading aerospace organisations from across Europe. It aims to develop satellite data-chain technologies for future Earth Observation and Telecommunication systems. The data chain elements are interconnected via a SpFi network.

Fig. 11 shows a photograph of the STAR-Tiger SpFi routing switch. STAR-Tiger is used in the Hi-SIDE project for transferring data at high data-rates between instruments, mass-memory, data compressor/processor and downlink transmitters. It is also used to provide the control network used by the control computer to control both the network and the equipment attached to the network. It has the following key features:

- 10 SpaceFibre ports
  - Two quad-lane ports
  - Eight dual-lane ports
  - Lane speed up to 6.25 Gbit/s
  - Port data rate 19.2 Gbit/s (quad-lane) and 9.6 Gbit/s (dual-lane port)
- 2 SpW ports
- 2 further SpW ports for programming STAR-Tiger
- Power consumption 13.5W typical at 20 °C
- 108 x 108 x 70 mm (excluding mounting brackets)
- Spaceflight TRL5/6 level design
  - Electronic components are EM flight parts or industrial/commercial equivalents of flight parts
  - Conduction cooled
  - Operating temperature range: -25 to +55 °C

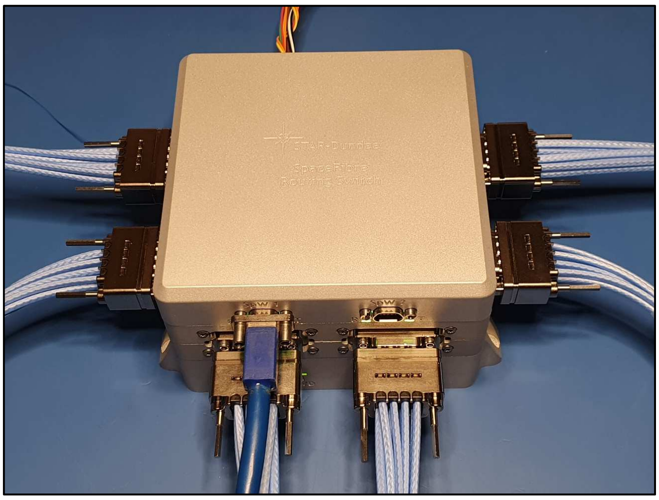


Fig. 11. Photograph of a STAR-Tiger unit.

VII. Conclusions

The STAR-Dundee SpFi Multi-Lane Routing Switch IP has been designed for ease of use and to achieve the highest lane rates on space-grade FPGAs, independently on the number of ports, virtual channels and number of lanes. Synthesis results show that area resources also scale well when the values of these configuration parameters are increased.

The Routing Switch supports lane rates of 3.125 Gbps in RTG4 and 6.25 Gbps in PolarFire FPGAs. In UltraScale and Versal Xilinx devices, faster speeds are possible. The multi-lane capability multiplies the data throughput of SpFi ports and provides hot and warm redundancy, or graceful degradation of the link bandwidth when no redundant lanes are available. It allows to implement a SpFi routing switch with more than 250 Gbps of aggregated bandwidth.

The Routing Switch also provides additional functionality to easily integrate SpaceWire devices into SpaceFibre networks and to distribute accurate time information across the network using broadcast messages.

The Routing Switch has been tested and subsequently validated within a full satellite data-chain technology demonstrator. Both commercial and a radiation-tolerant FPGAs have been used for these validation activities, ensuring full compatibility and defining an easy adoption path for this technology.

STAR-Dundee has developed and demonstrated the critical SpaceFibre Router technology necessary for cutting-edge on-board data-handling systems with very high data rate sensor and telecommunications systems.

Acknowledgment

The Hi-SIDE project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 776151.

References

[1] ECSS Standard ECSS-E-ST-50-11C, “SpaceFibre – Very high-speed serial link”, European Cooperation for Space Data Standardization, 15th May 2020. Available from http://www.ecss.nl.

[2] ECSS Standard ECSS-E-ST-50-12C, “SpaceWire, Links, Nodes, Routers and Networks”, Issue 1, European Cooperation for Space Data Standardization, July 2008, available from http://www.ecss.nl.

[3] S. Parkes, A. F. Florit, A. G. Villafranca, C. McClements and D. McLaren, "SpaceFibre network and routing switch," 2017 IEEE Aerospace Conference, 2017, pp. 1-7, doi: 10.1109/AERO.2017.7943805.

[4] A. Ferrer Florit, A. Gonzalez Villafranca, M. Farras Casas and S. Parkes, “SpaceFibre Routing Switch IP Implementation in Radiation-Tolerant FPGAs”, DASIA 2019

[5] ECSS Standard ECSSE-ST-50-52C, “SpaceWire – Remote memory access protocol”, European Cooperation for Space Data Standardization, February 2010, available from http://www.ecss.nl.

[6] Atmel, “AT7910E SpW-10X SpaceWire Router Datasheet”, Atmel, 2012, available from http://www.microchip.com.

[7] S. Parkes et al., “SpaceFibre Payload Data-Handling Network”, 2022 International SpaceWire Conference.

Wednesday 19th October

Components (Short)

NXS Series Interconnect System – Technical Attributes

Daniel A. Neary
Sr. Engineering Manager – Mil/ARINC
Smiths Interconnect
Kansas City, USA
daniel.a.neary@smithsinterconnect.com

Gregory Kay
Product Line Manager - Mil/ARINC
Smiths Interconnect
Elstree, UK
gregory.kay@smithsinterconnect.com

John Anderson
Principal Engineer – Mil Arinc
Smiths Interconnect
Elstree, UK
john.anderson@smithsinterconnect.com

Steven Douglas
Design Engineer
Smiths Interconnect
Elstree, UK
steven.douglas@smithsinterconnect.com

Robert Friedt
Electrical Engineer
Smiths Interconnect
Kansas City, USA
robert.friedt@smithsinterconnect.com

Space architectures are continuing to follow the trend of either daughter card to backplane or point to point system designs. The highest degree of flexibility is typically achieved using a point-to-point approach as seen in frame slice and SpaceWire or SpaceFibre architectures. SpaceFibre is a multi-Gbits/s, on-board network technology for spaceflight applications, which runs over electrical or fibre-optic cables.

There is an ever-increasing demand for rugged, high speed (up to 50 Gbps) connections, with the combination of controlled impedance differential pair signals together with the ability to add customizable modules, all with solderless board mounting. This technology will revolutionize the way in which space craft are designed and built by providing an unparalleled level of signal density and flexibility. The satellite market is moving away from RF Analog based payloads providing low speed telecommunication signaling, to a new Digital Transparent Processor architecture for high throughput satellites. Smiths Interconnect has developed, tested and qualified such a connector. This connector labeled NXS, is an advanced high speed interconnect solution to offer next generation data on demand, meeting both point to point and backplane connector requirements. The system has proven to be able to withstand space application requirements, including extreme levels of vibration, shock and climatic testing and offers a reliable way to implement high density interconnections with high-speed signal transmission requirements. The presentation will show what the next generation system looks like for daughter card to backplane and point to point system designs. It will explain how high data rate and high speed can be incorporated into one modular ultra-flexible system. The presentation will provide an answer to how extremely high EMI attenuation can be achieved in a modular interconnect system, while meeting rigorous shock and vibration performance levels.

Keywords— Smiths Interconnect, Connector for high-speed backplanes, space wire, space fibre, solderless PCB mound, composite shell, high contact density.

INTRODUCTION

The NXS Series is a space qualified high performance Interconnect System offering high data transmission capability in a solderless high density light weight modular configuration, primarily intended for satellite and space applications and supporting ESCC, ECSS and other demands. Using controlled impedance differential pair signals, each quad module comprising two differential pairs achieves digital data transmission rates up to 50 Gbps.

Physical Configuration

NXS in both 4 and 12 bay format supports cable to PCB or flexi-rigid interconnect, enabling various configurations suitable for Spacewire and other architecture including daughtercard to backplane or module to module arrangements. The receptacle can form part of a daughtercard assembly, or can be mounted to the equipment panel or chassis in an input/output or point to point application. Savers are used to facilitate test and integration activities without causing wear or damage to connectors. These are removed at the final stage of system build.

Fig 1 – Typical 4 bay receptacle and cabled plug

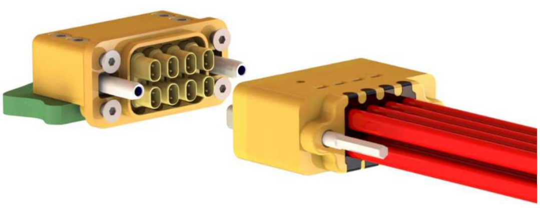
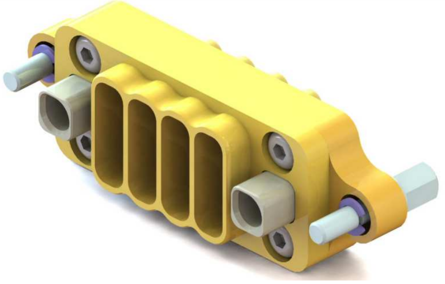


Fig 2 – 4 bay plug Saver

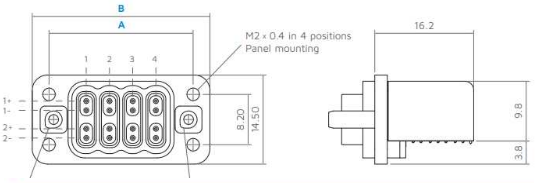


Connector Dimensions

NXS has been designed to have compact dimensions for optimal signal density, whilst also being compatible with typical shielded twinax cable, PCB layout design and isolation between modules.

Figs 3 and 4 show the outline dimensions for the 4 and 12 bay connectors.

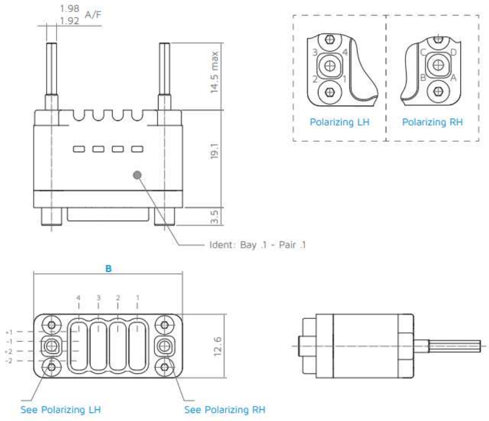
Fig 3 – NXS Receptacle Dimensions



Number of ways	A	B
4	23.6	29.1
12	54.0	59.5

Dimensions are in mm.

Fig 4 – NXS Plug Dimensions



Performance Summary

NXS is designed to offer balanced performance characteristics meeting the demands of data transmission in the space industry. Fig 5 shows the main performance parameters.

Fig 5 – NXS Performance Summary

Parameter	Level
Working Voltage	50 V RMS
Current	1 A
Date Transmission Rate	Up to 50 Gbps per channel
Impedance	100 Ω +/- 10%
Contact resistance	150 mΩ max
Insulation Resistance	1 GΩ min
Operating Temperature	-40°C to +125°C
Durability (saver fitted)	500 cycles

Mass

Low mass is a key driver in the design of NXS. Carbon reinforced PEEK composite polymer with gold over nickel plating is used for the main housings of the connector and titanium is used for hardware components, keeping the mass to a minimum whilst achieving other requirements including mechanical strength and outgassing.

A simple weight comparison between NXS and another interconnect system is shown below.

Description	Total Mass	Total Mass Savings
4 Way Space Fibre Type A, 1 metre cable	407.2 g	
NXS Composite 4 way Dual Twinax, 1 metre cable	117.5 g	290 g/mated pair

Modularity

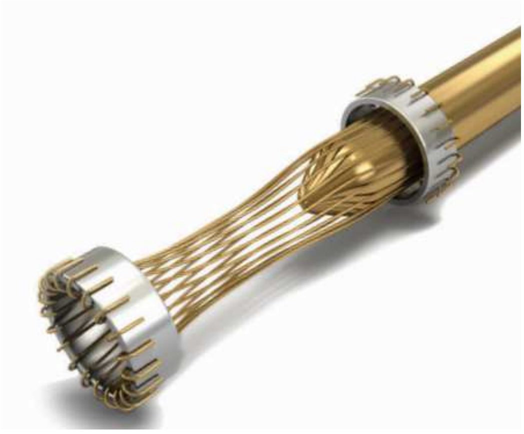
The 4 and 12 bay NXS connectors enable flexible and configurable system arrangements to be accommodated, with the added potential for mixing different contact or cable types in one connector housing. Connector bays can use one or two twinax differential pairs cabled as required, and other insert types such as low level power are planned for development.

Hyperboloid Contact Technology

NXS uses the proven and well known Hyperboloid contact at the interface, ensuring the highest level of performance and reliability for critical applications. Multiple linear contact paths give low contact resistance which remains stable over the lifetime of the equipment, critical for space applications. Along with long term reliability, the Hyperboloid contact has low insertion and extraction forces, giving high durability and long life, as well as an immunity to shock and vibration and the most demanding environments.

The NXS interface contact, named ‘Micro-boloid’ has a male pin of nominal diameter 0.4mm.

Fig 6 – Hyperboloid Contact



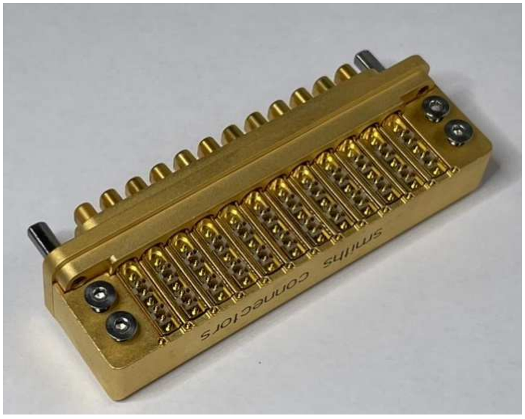
Solderless Termination

The NXS receptacle uses spring probe technology mating with gold plated target pads for the PCB interface, for differential pair contacts and ground contacts. This approach gives a number of benefits for system design, build integration, performance, test and repair.

Conventionally soldered PCB connectors can present significant challenges. The connector must either experience the PCB soldering reflow process, or be soldered after the main PCB assembly as a separate operation. Both of these situations make consistent achievement of the necessary quality requirements difficult and potentially costly.

Additionally, through-hole soldered connectors occupy a large area on both sides of the board, making this space unavailable for other componentry or circuit tracking, and soldered connectors are difficult or impractical to replace or rework. The use of spring probe contact technology removes all of these obstacles and enables a higher performing and simplified PCB design with the potential for solid ground planes to be used and no requirement for plated through holes.

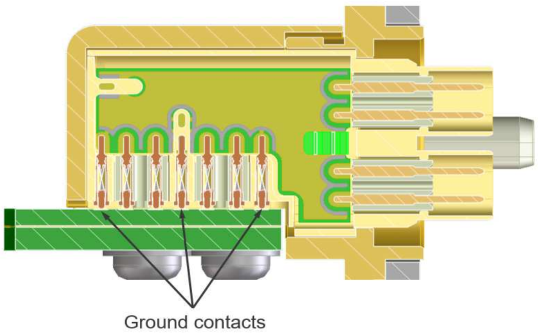
Fig 7 – Spring Probe Terminations



EMI Performance

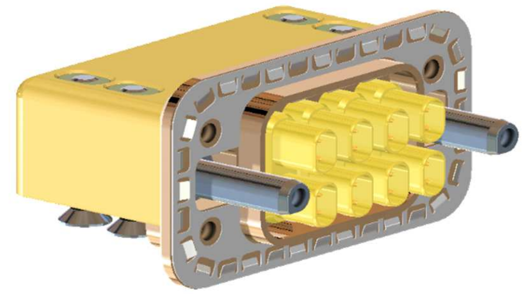
Each quadrax module comprising two differential pairs is fully shielded from end to end using gold plated metallic housings giving a high level of EMI immunity and channel isolation. The shielding and ground path extends from the twinax cable shield, through the plug and receptacle, to the multiple solderless spring probe ground contacts at the receptacle to PCB interface.

Fig 8 – NXS Receptacle cross-section



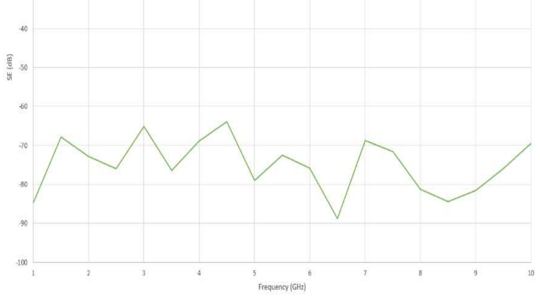
Where the NXS interface penetrates the housing of shielded equipment, a specifically designed EMI gasket combined with a metallic connector housing provides robust and reliable 360 degree shielding around the aperture.

Fig 9 – Receptacle with EMI Gasket



The EMI shielding effectiveness level achieved by this arrangement exceeds 65dB attenuation at 10 GHz.

Fig 10 – Shielding Effectiveness



Data Transmission Performance

The following shows some typical key data transmission test results, for one 100Ω differential pair.

(Each quadrax module contains two differential pairs).

Fig 11 – TDR Plot, Connector only

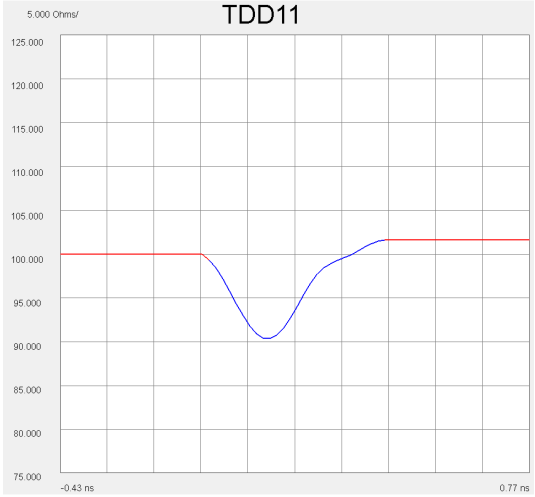
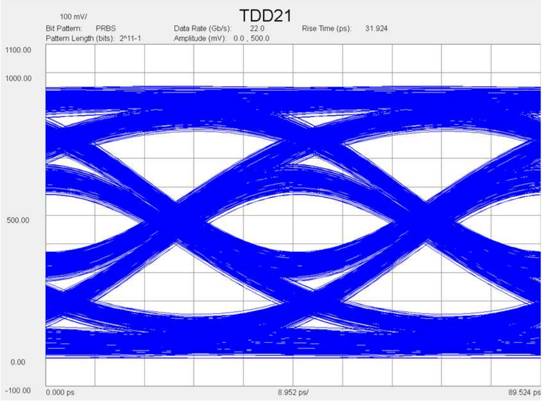


Figure 12 shows NXS connector eye diagram performance at 22 Gbps.

Fig 12 – Eye Diagram, 22 Gbps



Figures 13 shows eye diagram performance at 25 Gbps, including 3dB pre-emphasis - a technique commonly used to improve digital transmission signal quality

Fig 13 – Eye Diagram, 25 Gbps
3dB pre-emphasis

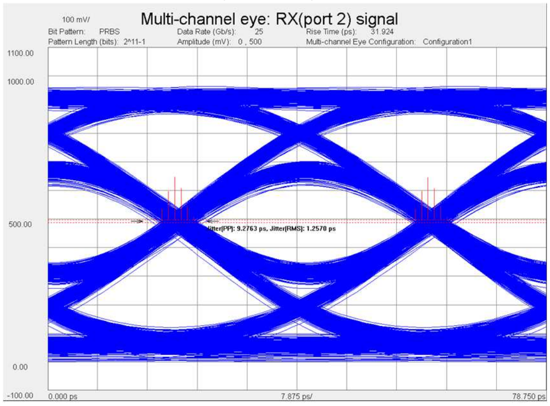
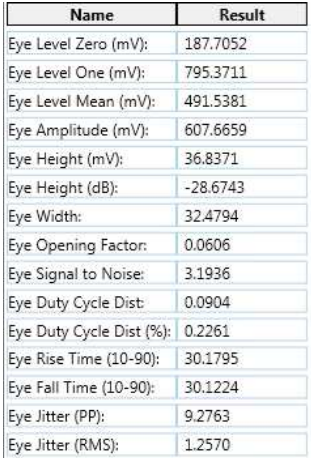


Fig 13a – Eye Diagram Data



Name	Result
Eye Level Zero (mV):	187.7052
Eye Level One (mV):	795.3711
Eye Level Mean (mV):	491.5381
Eye Amplitude (mV):	607.6659
Eye Height (mV):	36.8371
Eye Height (dB):	-28.6743
Eye Width:	32.4794
Eye Opening Factor:	0.0606
Eye Signal to Noise:	3.1936
Eye Duty Cycle Dist:	0.0904
Eye Duty Cycle Dist (%):	0.2261
Eye Rise Time (10-90):	30.1795
Eye Fall Time (10-90):	30.1224
Eye Jitter (PP):	9.2763
Eye Jitter (RMS):	1.2570

Figs 14 and 15 show insertion and return loss plots, indicating data transmission performance capability beyond 22Gbps.

Fig 14 – Insertion Loss, 22 GHz

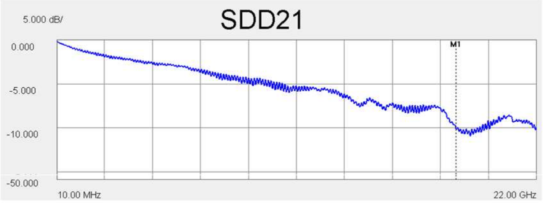
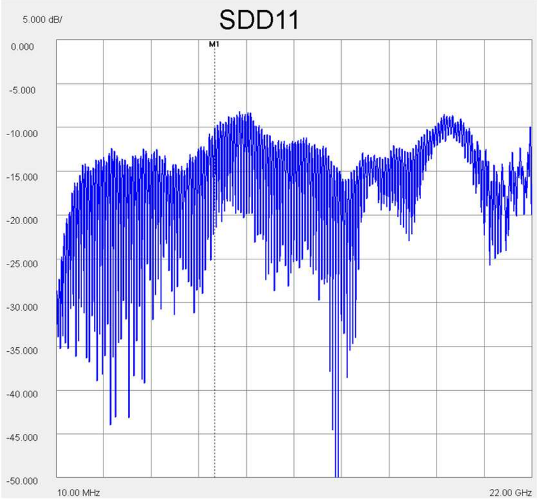


Fig 15 – Return Loss, 22 GHz



HIGH DATA RATE FLAT FLEXIBLE CABLE FOR SPACEWIRE & SPACEFIBRE

Provost Kévin
Reasearch and development

Axon' cable
Montmirail, France
k.provost@axon-cable.com

Hermant Stéphane
Reasearch and development

Axon' cable
Montmirail, France
s.hermant@axon-cable.com

Abstract— **Flat Flexible Cables have many advantages and among them, a very constant geometry and a robust assembly. Thanks to that it appears it would be interesting to take advantage of this technology to make very high speed connections through a very low SKEW of the electric signals using this media. Moreover FFCs must be shielded to guarantee the characteristic impedance of the transmission lines and also to offer an EMC protection of the link.**

In this paper, Axon' reviews high speed signal integrity and EMC results from the ongoing evaluation testing of a link using a new MicroMach® flat shape connector variant dedicated to shielded FFC. This connector is based on the design of the MicroMach® SpaceWire® connector ESCC3409/002.

I. Introduction

As a manufacturer of classic Micro-D connectors, of a wide variety of round cables and interconnect used in Space, Axon' Cable is very well versed in the pros and cons of harnesses used in harsh environment and particularly concerning EMC constraints.

Axon' is also leader for Flat Flexible Cable commonly used in consumer electronics, office automation and automotive applications. In previous SpaceWire conferences papers [1] [2] [3] [4]Axon' presented its developments in Low Mass SpaceWire cable, and also an overview of the new MicroMach® impedance-matched SpaceWire® connector development (ESCC3409/002 and ESCC3401/095 c.f. ESA contract N° 4000113741/15/NL/SW).

Flexible Flat Cables (FFC) are emerging in the space sector because they offer advantages versus round wires harness used until now. Compared to bundles of round cable and wires, their lightness and flexibility are elements that can solve complex situations. In addition, their much better heat dissipation allows higher currents to be passed for the same copper cross-section..

II. Design of the high data rate flat cable assembly

First prototype of a Flat Flexible SpaceWire® link has been assembled using shielded flat cable and a new, flat shaped connector derived from the MicroMach® design parts (see Figure 1). These connectors present 4 cavities adapted to 100Ω. This trial product opens the possibility of a new automated manufacturing solution improving flexibility and saving space. At the same time the media is improved in skew and characteristic impedance regularity features.

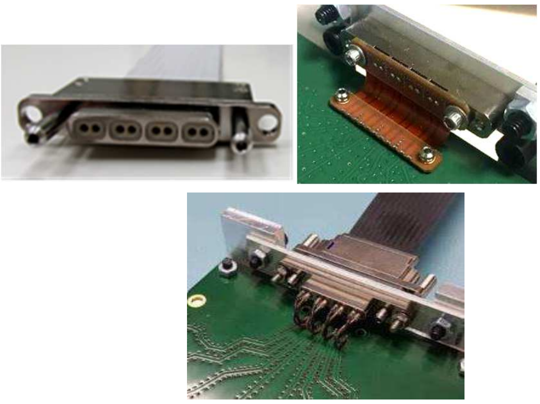


Figure 1. *Flat shaped MicroMach® connectors (inline & PCB variants)*

A. The cable

The media is based on 11 tracks of AWG29 flat conductors (0.7mm*0.1mm=0.07mm²), 4 differential 100Ω channels separated by 1track and 360° shielding. Overall dimension is around 1.3mm * 21mm (See figure 2) Signal transmission is routed on four differential stripline structure (see Figure 3) separated by one ground track. In figure 4, a picture of a LowMass SpaceWire cable (AWG28) with the FFC.

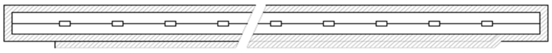


Figure 2. *Flat Flexible Cable section*

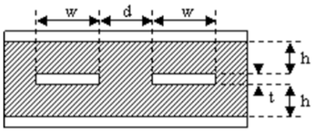


Figure 3. *Differential stripline structure*

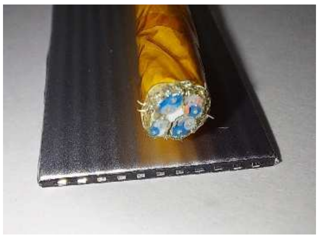


Figure 4. *FFC cable versus standard SpW cables*

In table 1 we compare the main dimensions and weight between the LowMass SpaceWire cable and the FFC.

	ESCC390200401 (LMSpW AWG28)	FFC
Mass	42g/m	34g/m
Height	6.5mm	1.3mm
Width	6.5mm	21mm

Table 1. Table with main characteristics comparison between these new FF cable versus ESCC390200401 (LowMass SpW cable)

B. The connector (Flat MicroMach® type)

The connector, which is a design heritage from MicroMach® but adapted to the FFC shape and dimensions is base on the following parts (see ***Figure 5***):

- MicroMach® (4x 100Ω differential cavities with PEEK inserts)
- TwistPin contacts technology (used on MicroD connectors)
- Track bonding with soldering or welding method (***Figure** 7*)
- Shield bonding with EMC gasket compression
- Guide pins to ease and secure the mating
- Hardware to lock the connection between plug and socket

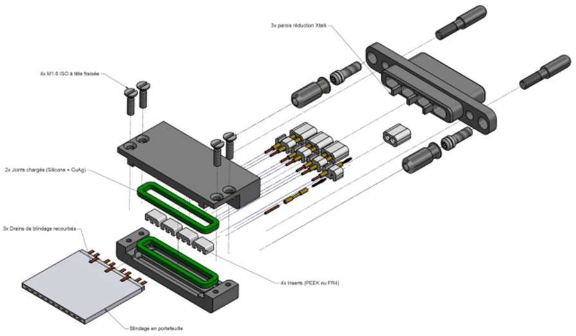

3x parois réduction Xtalk
4x M1.6 ISO à tête fraisée
2x Joints chargés (Silicone + CuAg)
3x Drains de blindage recourbés
4x Inserts (PEEK ou FR4)
Blindage en portefeuille

***Figure 5**. Inline connector parts overview*

From equipment side, PCB variant could be bonded on PCB with standard method using matched impedance unshielded pair but also with flex-rigid PCB to ease the bonding.

This flex-rigid PCB could be mount by soldering or using dedicated interposer then having a fully dismountable termination. (See ***Figure 6***)

Interposer contacts are made with dedicated twistpin contacts to insure several contact points between the PCB and the flex.

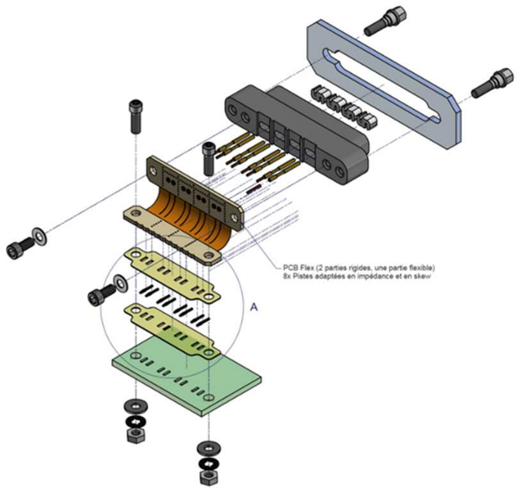

PCB Flex (2 parties rigides, une partie flexible)
8x Pistes adaptées en impédance et en skew
A

***Figure 6**. PCB-Flex connector parts overview*

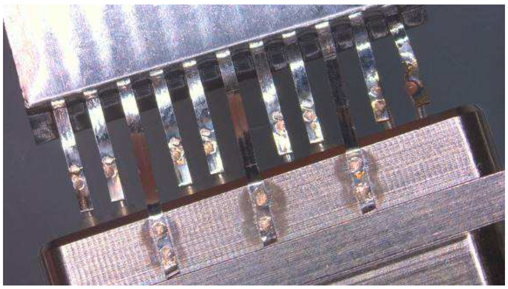


***Figure** 7. Process compatible with electric welding*

Direct connection to PCB is also in study to reduce the number of transitions using only an interposer contact between the cable and the PCB tracks.

EMC improvement is foreseen between connector's couple using "EMI clip" on male (see ***Figure 8***) or female connector . The qualification of this technology is ongoing on microD, MicroD fast lock, MicroD combo (Versatys®) and could be integrated on MicroMach® connectors.

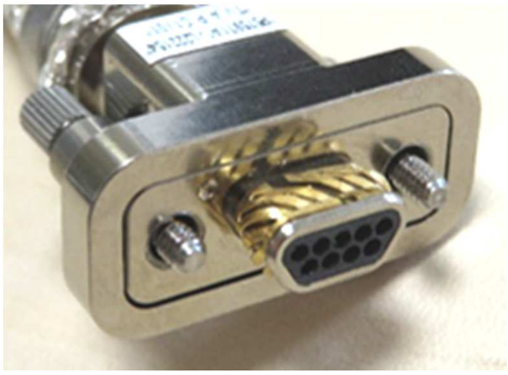


***Figure 8**. Example of EMI clip integrated in a MD9 male connector*

III. TEST VEHICLES MEASUREMENT RESULTS

The test vehicle on which the measurements have been performed is a 2m long Flat Flexible Cable with AWG28 flat conductors linked to 2 adapted MicroMach® flat variant connectors.

The test jigs were made with connectors counterparts terminated with flex connected to PCB then equipped with SMA connectors to be compatible to measurement equipment (one differential transmission line is linked to 2 coaxial cables. see ***Figure 9***)

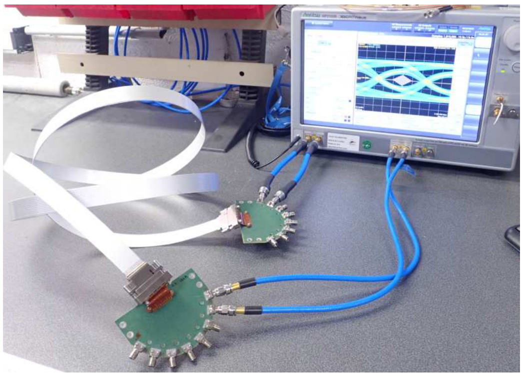


Figure 9. *Flat shaped MicroMach® test vehicle with PCB test jigs & SMA test leads*

A. Signal integrity test results

Signal integrity measurements has been performed with Eye pattern transmission for SpaceWire up to 3.4Gb/s (See figure 10) and Space Fibre signal up to 6.25Gb/s (See figure 11) thanks to mainly constant characteristic impedance and low skew cable.

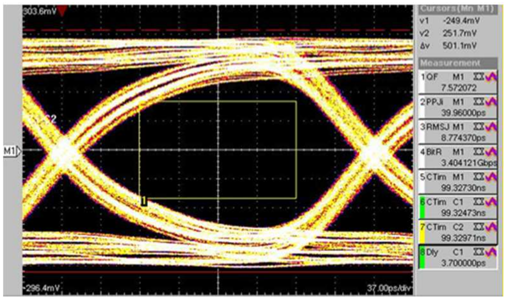


Figure 10. *Eye pattern at 3.4GB/s for a 2 m long assembly*

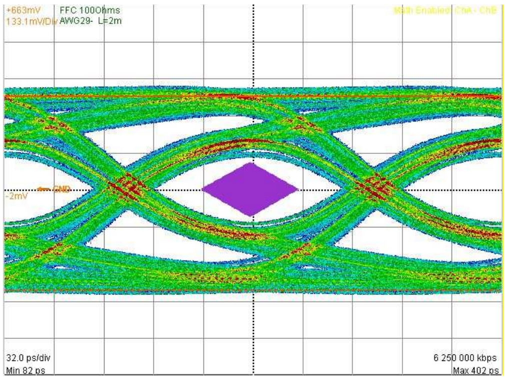


Figure 11. *Eye pattern at 6.25GB/s for a 2 m long assembly including PCB connectors & interposer (SpaceFibre mask)*

- Skew measured on this harness was less than 5ps. (quite low for 2m length link)
- Characteristic impedance was measured at 103 Ω with a peak to peak ripple of less than 2 Ω (see graph in ***figure12***).

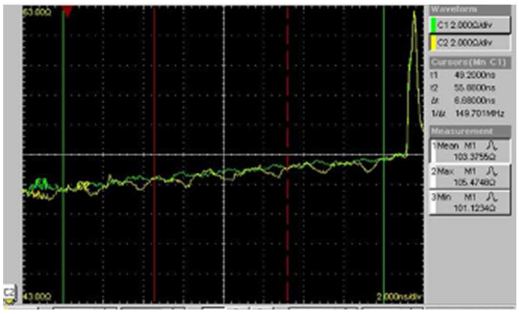


Figure 12. *Characteristic Impedance in reflectometry*

Furthermore, validation has been performed on a link connected to 2 PCB test jigs with a LAN tester overpassing TIA CAT6A ETHERNET channel limits. This means working up to 10Gb/s on Ethernet networks.

The limitation in frequency & data rate is mainly coming from cable losses (in the dielectric & in the conductors) as skew is strongly reduced compare to twisted pair it could not be part of the budget up to around 10Gb/s.

If we compare the insertion losses of the FFC (AWG29) regarding the SpW cable, we see the losses are next to the limits of the SpaceWire AWG26. (see the chart in figure 13)

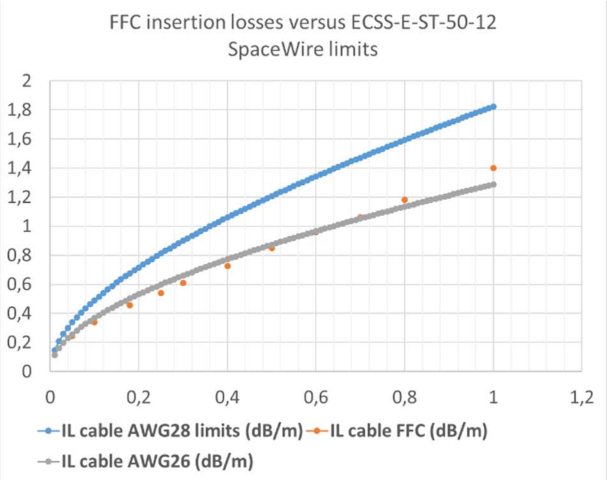


Figure 13. Comparison between losses on SpaceWire cables versus FFC cable.

B. *C.E.M. : Shielding effectiveness in Stirred Mode chamber. Comparison with round cable versions.*

Shielding Effectiveness is measured on 3 models in the axon' Stirred Mode Chamber up to 18Ghz:

- 2m of low Mass SpaceWire cable connected on 2 MicroMach connectors.
- 2m of Flat Flexible Cable connected on 2 special flat MicroMach® connectors.
- 2m of low Mass SpaceWire cable connected on 2 MicroD connectors.

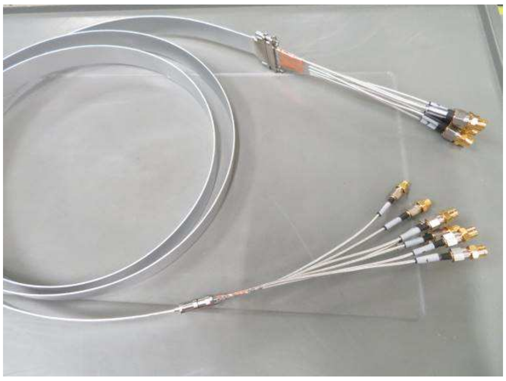


Figure 14. *Flat shaped MicroMach® test vehicle with SMA test leads*

The results are quite encouraging for FFC harness which is almost as good as the well-known SpaceWire harness. (See ***figure 15***).

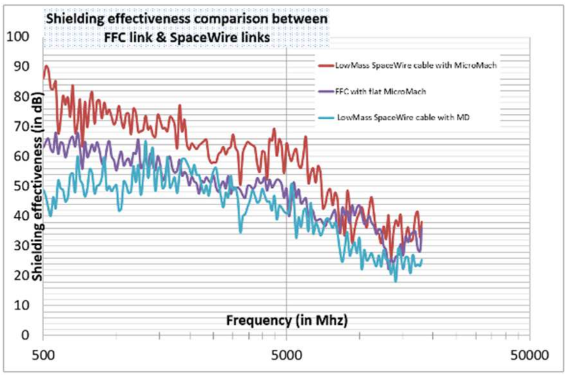


Figure 15. *Measurement in stirred mode chamber of 3 harnesses technology*

C. *Routing of HDR FFC assemblies.*

Of course flat cables have a natural bending direction which allows interesting dynamic applications. But when it is necessary to go in a radial direction to the cable, the FFC must be bent. This operation is done with very light variation of the performances thanks to the common solid dielectric.

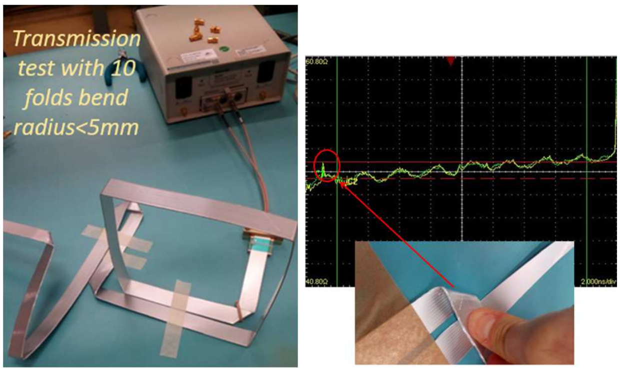


Figure 16. *Characteristic Impedance in reflectometry*

Conclusion

Flat cables offer an interesting alternative in terms of mass and space, whether they are designed to carry power, analog signals, digital signals or even very high speed signals. Axon has built FFCs to support high data rates of several Gb/s so that they can be used for protocols such as SpaceWire, SpaceFibre or TTEthernet .

-The construction of FFCs allows transmission lines with very low intra and inter skews, hardly measurable.

-The shielding of these FFCs can be done on 360° and thus bring a powerful electromagnetic protection better than the results from the existing SpaceWire link with MicroD 9ways connectors.

Moreover, we have proven that routing the FFC with very low bend radius along the satellite does not degrade its performances.

Up to now, the cable service temperature is limited to 80°C due to the thermal characteristic of the dielectric. A new study is ongoing to replace it in order to improve this feature.

In the same time, axon' is working on ZIF/LIF connector on PCB and will study the possibility to support high data rate FFC media with this technology.

Studies and manufacturing improvements are ongoing to propose also power shielded links, means of fixations in severe environments and connections parts for harness and equipment.

References

[1] Stéphane Hermant,Kevin Provost, Guillaume Frenoi, Gilles Rouchaud, Frederic Lescoat "High data rate Flat flexible cable (FFC) - shielding EMC performances", 2022 ESA WORKSHOP ON AEROSPACE EMC, May, 2022.

[2] Stéphane Hermant, "Compact Impedance Matched Connectors for SpaceWire", DASIA2021, October, 2021.

[3] Stéphane Hermant, Nigel Kellet, "The evolution of SpaceWire interconnect," in Proc. SpaceWire International Symposium , Longbeach, USA Ca, May, 2018.

[4] Stéphane Hermant, Kevin Enouf, "SpaceWire connector development: MicroMach SpaceWire," in Proc. Seventh International SpaceWire Conference, Yokohama, JAPAN, October, 2016.

A low-overhead AXI bridge on SpaceWire for multi-device spacecraft systems

Claudio Rubattu
SITAEL S.p.A.
56121 Pisa, Italy
claudio.rubattu@sitael.com

Walter Errico
SITAEL S.p.A.
56121 Pisa, Italy
walter.errico@sitael.com

Kostas Marinis
ESA - ESTEC
2201 Noordwijk, Netherlands
kostas.marinis@esa.int

***Abstract*—The increasing complexity of spacecraft on-board computational capability demands heterogeneous architectures with various devices (e.g., single-chip systems, or boards, responsible for processing and reconfiguration managers) composed of several elements (e.g., processors, memories, interconnects, interfaces, etc.). Such architectures consist of independent devices that rely on specific protocols to exchange data and configuration information with each other: a proper candidate is the SpaceWire (SpW) protocol. Nevertheless, compatibility between the SpW and the on-chip communication standards used for accessing architectural elements (deployed inside and/or outside the devices) must be guaranteed. In particular, when the architecture involves elements with interfaces based on non-reduced protocols (e.g., with handshake mechanisms and burst support), an implementation from scratch may be required in order to meet the project requirements (e.g., resource utilization, bandwidth and latency). A common case is related to interfacing the SpW with a popular on-chip protocol: the AMBA AXI4 memory-mapped standard.**

In this paper, a design of coupled interface set capable of bridging AXI4 memory-mapped requests in a SpW-based multi-device system is proposed. This set, called "AXI-SpW-AXI Bridge", has been implemented as synthesizable RTL and is composed of an AXI-to-SpW Bridge connected with an SpW-to-AXI Bridge via SpW protocol. Both bridges consist of i) a finite state machine, which manages its AXI interface (master or slave respectively), and ii) an SpW IP, for their communication with the SpW standard. In addition, the proposed bridge does not include the functions of the RMAP standard for memory interfacing, in order to limit the typical overheads associated with its implementation. The AXI-SpW-AXI Bridge verification has been performed by means of its integration into a Xilinx Zynq-7000 SoC and a Microchip Polarfire FPGA: the stimulus generation comes from the processing system, which is linked to the AXI-to-SpW bridge through interconnects and DMA also deployed in the Zynq programmable logic, while the SpW-to-AXI Bridge is mapped onto the PolarFire fabric. A post-synthesis evaluation of the AXI-SpW-AXI Bridge has shown a low resource utilization of both use-case devices (less than 2% of lookup tables and 1% of registers), and an almost equal distribution of resources in the two internal bridges. Compared to the state-of-the-art RMAP-based IP Cores, these results suggest that the proposed AXI-SpW-AXI Bridge represents a valuable trade-off when latency and resource constraints are more restrictive than the functional requirements of the on-chip protocol.

***Index Terms*—SpaceWire, AXI, SoC, FPGA, Spacecraft On-Board Data-Handling**

I. Context, Background and Motivation

In the context of space applications, Field-Programmable Gate Arrays (FPGAs) have assumed an important role in on-board electronics [1]. These platforms represent a trade-off between general-purpose and specialized solutions. In particular, they are more efficient (in terms of timing, power consumption, and circuitry area) compared to general-purpose solutions, and have a higher flexibility (in terms of cost, design effort, and functional reconfiguration) than the Application-Specific Integrated Circuits (ASICs) [2]. Among the various on-chip communication standards typically adopted in space-related FPGAs, this paper focuses on AXI4 memory-mapped and SpaceWire (SpW), and especially on their combined use. The Advanced eXtensible Interface (AXI), developed by ARM, is certainly one of the most widely used on-chip communication protocols in FPGAs [3]–[5], thanks to the support offered by the main FPGA vendors (such as AMD Xilinx, Intel and Microchip), which often integrate ARM processors into their devices. The AXI protocol is based on 5 channels (i.e., sets of signals): 2 of which are dedicated to read transactions (read address and read data) and 3 to write transactions (write address, write data, and write response) [6]. In addition, the protocol specifies 3 types of interfaces: i) the AXI-Full for memory-mapped data, ii) the AXI-Lite for handling control registers, and iii) the AXI-Stream for streaming data. In this work, the AXI-Full has been chosen for its higher flexibility, which enables burst-based and register-like access. On the other hand, the SpaceWire (SpW) [7] is a full-duplex serial interface standard created in order to simplify data exchange in the increasingly heterogeneous architectures used in the space domain, and whose development is coordinated by the European Space Agency (ESA). The connection between different architectural elements via SpW requires that they are provided with an SpW Intellectual Property (IP) or Core. As depicted in Figure 1, an SpW Core typically consists of an Finite State Machine (FSM) that manages the data streams in transmission (TX) and reception (RX), associated with the serial (SpW) and parallel (SpW Core Link) interfaces. Nevertheless, to perform read and write operations in memory, the addition of circuitry and the extra protocol management are required to support the SpW Core [8]–[13], as defined in the Remote Memory Access Protocol (RMAP) standard [14]. An RMAP-based transaction generally involves sending a command (with a specific format with respect to the type of memory access) and optionally receiving a reply packet. As a consequence, implementing the RMAP over the SpW link leads to higher utilization of FPGA resources, especially flip flops (FFs) and look-up tables (LUTs), because of the hardware blocks to be included as a front-end of the parallel interfaces. In addition, the protocol-related overhead degrades latency performance, since it implies a higher complexity of operations to be accomplished during the data transfer. To the authors' knowledge, there are no available SpW Cores capable of interfacing with memory nodes via the AXI4 memory-mapped protocol without the use of the RMAP standard. For these reasons, this paper evaluates the implementation of a bridge combination, called AXI-SpW-AXI bridge, to handle access to a SpW memory node by another SpW node. Specifically, the two evaluated SpW nodes correspond to two systems which are composed of hardware blocks communicating with each other via AXI, and are embedded in two different devices, a Xilinx System on Chip (SoC) and a Microchip FPGA.

The rest of the paper is organized as follows. Section II describes the high-level structure of the proposed bridge and the handling strategy used for the AXI transfers on the SpW link. In Section III, results in terms of hardware resources and latency are presented. Finally, Section IV concludes the paper.

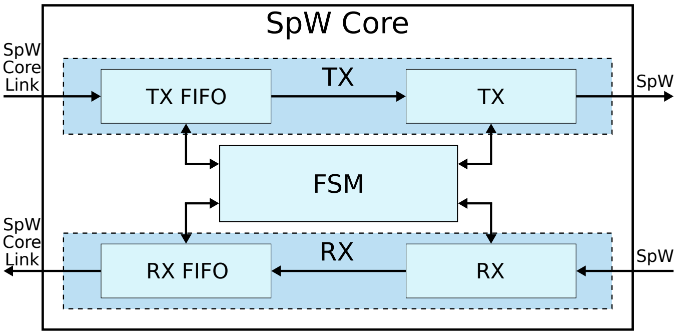


Fig. 1. Block diagram of the SpW Core. Incoming and outgoing arrows represent the presence of slave and master interfaces, respectively.

II. AXI-SpW-AXI Bridge

The AXI-SpW-AXI Bridge enables transfers requested from an AXI4 memory-mapped slave interface to an AXI4 memory-mapped master interface, communicating with each other via SpW protocol. In Figure 2, the block diagram is depicted in hierarchical terms. The AXI-SpW-AXI Bridge communicates with the outside with AXI4 memory-mapped protocol via two interfaces (one slave and one master). These interfaces are actually managed by two internal blocks (AXI-to-SpW Bridge and SpW-to-AXI Bridge respectively), communicating with each other via SpW protocol. In turn, the two mentioned blocks consist of a SpW Core and an FSM, which exchange data via a parallel protocol defined by the SpW Core Link. The SpW Core instantiated in both blocks are identical. While their FSMs (respectively AXI-to-SpW FSM and SpW-to-AXI FSM) actually manage the AXI interfaces present in the IPs, the slave and the master respectively. The internal states of the FSMs are grouped according to type (read or write) and channel (address, data and response) of the corresponding AXI transaction (to match with the AXI standard [6]). Since the AXI-SpW-AXI Bridge is only capable of handling one AXI request at a time, it must be completed before the next one. Regarding the strategy used for read and write transactions, this is discussed in the rest of this section (see Figure 3).

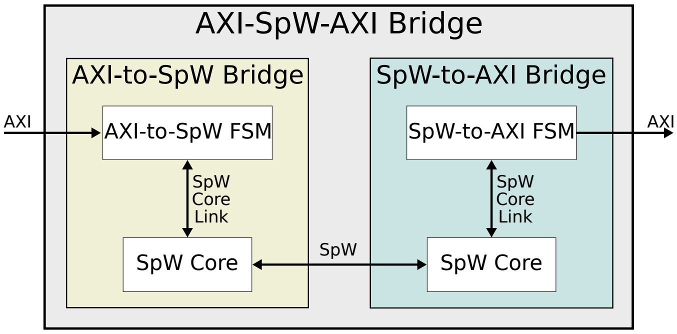


Fig. 2. Block diagram of the AXI-SpW-AXI Bridge. The clock and reset signals have been omitted for simplicity; and incoming and outgoing arrows represent the presence of slave and master interfaces, respectively.

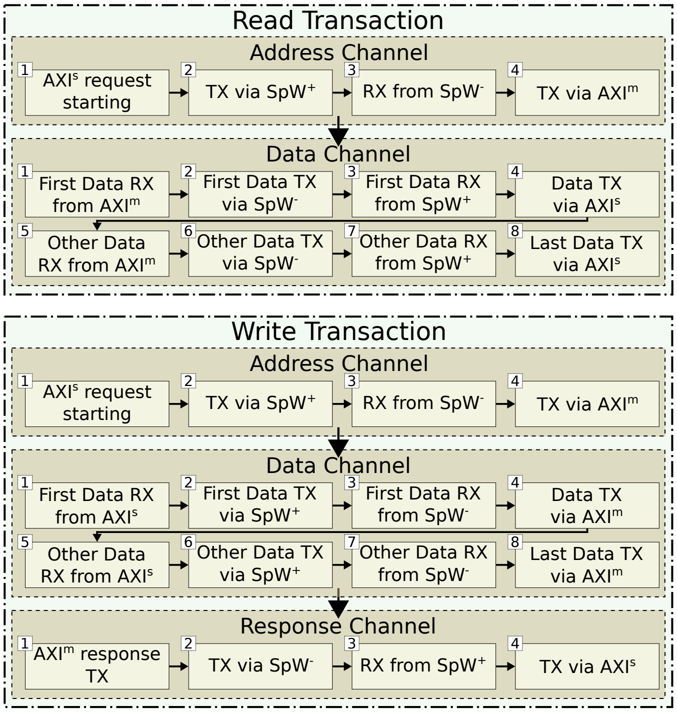

Read Transaction
Address Channel
1 AXI^s request starting
2 TX via SpW$^+$
3 RX from SpW$^-$
4 TX via AXI^m
Data Channel
1 First Data RX from AXI^m
2 First Data TX via SpW$^-$
3 First Data RX from SpW$^+$
4 Data TX via AXI^s
5 Other Data RX from AXI^m
6 Other Data TX via SpW$^-$
7 Other Data RX from SpW$^+$
8 Last Data TX via AXI^s
Write Transaction
Address Channel
1 AXI^s request starting
2 TX via SpW$^+$
3 RX from SpW$^-$
4 TX via AXI^m
Data Channel
1 First Data RX from AXI^s
2 First Data TX via SpW$^+$
3 First Data RX from SpW$^-$
4 Data TX via AXI^m
5 Other Data RX from AXI^s
6 Other Data TX via SpW$^+$
7 Other Data RX from SpW$^-$
8 Last Data TX via AXI^m
Response Channel
1 AXI^m response TX
2 TX via SpW$^-$
3 RX from SpW$^+$
4 TX via AXI^s

Fig. 3. Read and write transaction steps depending on the channel. Superscripts s/+ and m/- indicate the interfaces of AXI-to-SpW and SpW-to-AXI bridges, respectively.

A. Read Transaction

1) Address Channel: Composed of 4 main steps, the first phase of a read consists in sending transaction properties (e.g. target address, transfer length, burst type) from the AXI signals associated with the address channel. In the first step, the mentioned properties are stored into the registers of the AXI-to-SpW FSM, after the assertion of the valid signal of the AXI slave interface. In the second step, the request is forwarded towards the other SpW-to-AXI FSM through the SpW Core, which sends: i) a control SpW packet, ii) all the request properties, iii) an End Of Packet (EOP) and iv) a CRC packet. In the third step, these are received by the other SpW Core and further forwarded to the SpW-to-AXI FSM, which performs a CRC check at the end of the packet stream. The last step concerns the read request forwarding via the AXI master interface.

2) Data Channel: The second (and last) phase of a read transaction involves the AXI signals associated with the data channel, and consists of 8 main steps. In the first step, data are stored into the registers of the SpW-to-AXI FSM, after the assertion of the valid signal of the AXI master interface. In the second step, the request is forwarded towards the other AXI-to-SpW FSM through the SpW Core, which sends: i) a control SpW packet, ii) read data and iii) a CRC packet. In the third step, these are received by the other SpW Core and further forwarded to the AXI-to-SpW FSM, which performs a CRC check at the end of the packet stream. The fourth step is the forwarding of the read data and the request response towards the AXI slave interface. The step 5 is equal to the first one, except for the FSM handling when it ends. Also, the step 6 corresponds to the step 2, except for the absence of the control package and, only in case of last data, the adding of the EOP. In step 7, the same operations as in step 3 are addressed, except for expecting only data packet (the control packet has been received in step 3) and, only in case of last data, the EOP. The last step is equal to the step 4, except for the assertion of the last data signal.

B. Write Transaction

1) Address Channel: The first phase of a write transaction is handled by the AXI signals related to the address channel. Since this phase is practically identical to its equivalent already described for the read transaction (see Section II-A1), the description of the 4 steps is avoided in this section. Nevertheless, the only different notes concern the handling of: i) the AXI-to-SpW FSM in the step 3 to properly manage the address and data channels with an operation of synchronization; and ii) the SpW-to-AXI FSM after the step 4 to wait for a request related to the data channel expected by the SpW Core Link.

2) Data Channel: Since the second phase of a write transaction correspond to its counterpart illustrated in Section II-A2, the 8 steps are not described in this section. However, there are some variations to highlight. Indeed, the data flow is in the opposite direction, that is from the AXI slave interface to the AXI master interface. Moreover, the presence of a strobe signal instead of the response signal, since a dedicated response channel is required. Finally, the management of the SpW-to-AXI FSM in the step 8 to wait for the response associated with the write response channel expected by the AXI master interface.

3) Response Channel: The third (and last) phase of a write transaction involves the AXI signal of the response channel, and the same steps of the address channel. Nevertheless, also in this case, exceptions are presents: i) the packets are moved in the other direction (from the AXI master interface to the AXI slave interface); ii) the initial and final states of both FSMs (AXI-to-SpW FSM and SpW-to-AXI FSM), since the response channel concludes the write transition; and iii) the required information carried by the response signal.

III. Experimental Results

Figure 4 shows the system used for the assessment, which considers writing and consequent reading of one or more data using a memory controller. The target devices are the AMD Xilinx Zynq (XC7Z020CLG484-1) and the Microchip PolarFire (MPF300TS-1FCG1152EES), which are included in evaluation boards provided by the vendors (respectively [15] and [16]). For these targets, hardware synthesis was performed with the respective simulators: Xilinx Vivado ML v2021.2 and Microchip Libero SoC v2021.2. In the assessment, write and read requests are generated by the Zynq-7000 Processing System IP via specific functions provided by Xilinx [17]. In particular, the *write_data* (*write_burst*) and *read_data* (*read_burst*) functions have been used in the case of single data (burst) to be written and read respectively.

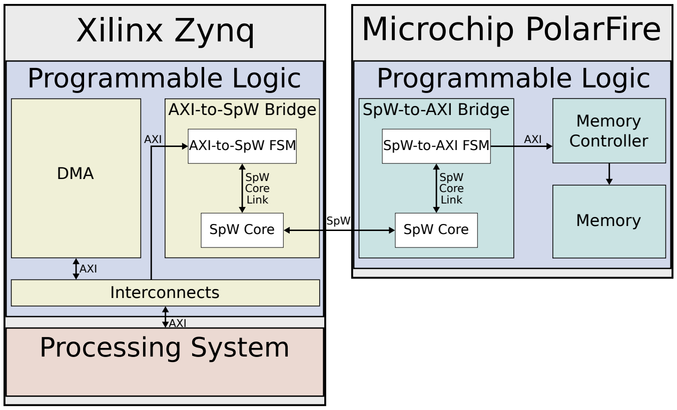


Fig. 4. Block diagram of the system used for the assessment of the AXI-SpW-AXI Bridge. Incoming and outgoing arrows represent the presence of slave and master interfaces, respectively.

A. Resource Usage Evaluation

In Figure 5, synthesis results associated with the bridges deployed onto the Xilinx and Microchip FPGAs are shown. The required resources are registers (REG) and logic elements (LOGIC) distributed approximately symmetrically between the two blocks. Indeed, the AXI-to-SpW (SpW-to-AXI) Bridge is synthesized with 771 FFs (621 DFF) and 936 LUTs (968 4LUTs) in the Xilinx (Microchip) target, which correspond to 0.72% (0.21%) and 1.78% (0.32%) of Xilinx (Microchip) FPGA utilization, respectively. These amounts of resources lead to an estimated power consumption of 3mW for both FPGA targets. Moreover, referring to works which reports synthesis results on FPGAs, the proposed implementation requires less usage of resources compared to the solutions presented in [9] (2416 registers, 3752 LUTs and 32 RAM blocks of the Xilinx Kintex 7-410T) and in [10] (2155 registers and 4217 LUTs of the Xilinx Virtex 5Q-XQ5VFX130T). Consequently, the usage of the proposed solution is at least 3 and 4 times lower in terms of FFs and LUTs, respectively. Although the actual gain may be considered as not justified in the chosen targets (on the order of 1% for FFs and 3%

for LUTs), this may become relevant in small FPGAs used in space (e.g. Microchip ProASIC3 FPGAs [18]).

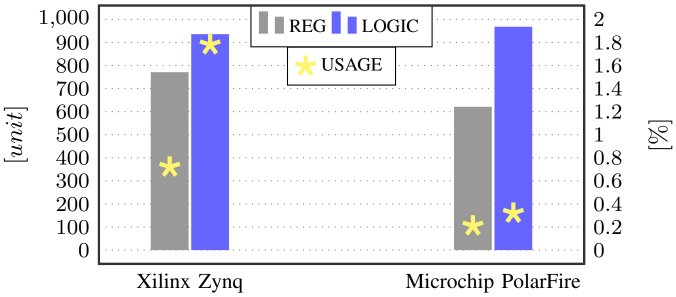


Fig. 5. Synthesis results on Xilinx and Microchip FPGAs, in terms of resources (registers: REG, and logic elements: LOGIC) and utilization. Xilinx/Microchip resources: REG=FF/DFF, LOGIC=LUT/4LUT.

B. Response Time Evaluation

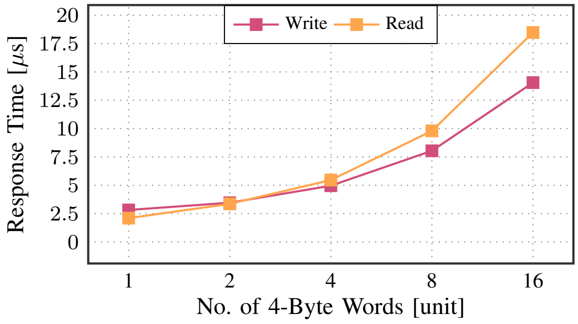


Fig. 6. Response time (with a system clock frequency of 50 MHz, and a SpW clock of 100 MHz) for writing and reading data of a variable number of 4-byte words from the Zynq-7000 Processing System IP perspective.

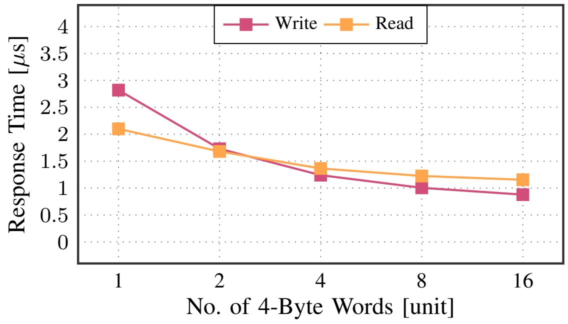


Fig. 7. Response time (with a system clock frequency of 50 MHz, and a SpW clock of 100 MHz) per single data written and read for a variable number of 4-byte words from the Zynq-7000 Processing System IP perspective.

Response times have been obtained from the time difference between the valid signals related to the address and response/data channels of the AXI slave interface. Figure 6 reports the response time for each of these write and read transactions. In particular, the time required for the single data item (in this case, a 4-byte word) decreases as the length of the burst increases (see Figure 7). For a write transaction, the response time for a single data ranges from 2.820 us to 0.879 us for a burst length of 1 and 16 respectively. On the other hand, the same time metric respectively ranges from 2.100 us to 1.155 us in case of read transaction. Focusing on the SpW link, the overhead of the RMAP command (16 bytes) affects latency more than that one of the proposed solution (9 bytes), especially when the data consists of a few bytes.

IV. Conclusion

In this paper, the AXI-SpW-AXI Bridge has been presented with the aim of connecting a memory and an IP (both capable of communicating via AXI) over SpW protocol without leveraging on the RMAP standard. The synthesis results of the proposed implementation show a lower demand for resources than the case with RMAP. In addition, the lower complexity of the used protocol leads to a lower latency overhead, especially in case of peer-to-peer exchange of a few data. Therefore, the solution provided identifies a different trade-off that may be convenient when latency and resource requirements are highly constrained.

References

[1] M. Wirthlin, "High-reliability fpga-based systems: Space, high-energy physics, and beyond," *Proceedings of the IEEE*, vol. 103, no. 3, pp. 379–389, 2015.

[2] A. Fuchs and D. Wentzlaff, "The accelerator wall: Limits of chip specialization," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2019, pp. 1–14.

[3] K. Vipin and S. A. Fahmy, "Fpga dynamic and partial reconfiguration: A survey of architectures, methods, and applications," *ACM Computing Surveys*, vol. 51, no. 4, 2018.

[4] F. Restuccia, M. Pagani, A. Biondi, M. Marinoni, and G. Buttazzo, "Is your bus arbiter really fair? restoring fairness in axi interconnects for fpga socs," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 5s, oct 2019.

[5] K. P. Seng, P. J. Lee, and L. M. Ang, "Embedded intelligence on fpga: Survey, applications and challenges," *Electronics*, vol. 10, no. 8, 2021.

[6] AMD Xilinx. AXI Reference Guide. https://docs.xilinx.com/v/u/14.1-English/ug761_axi_reference_guide.

[7] S. Parkes and P. Armbruster, "Spacewire: a spacecraft onboard network for real-time communications," in *14th IEEE-NPSS Real Time Conference*, 2005.

[8] ESA. SpaceWire-AMBA - HDL. https://www.esa.int/Enabling_Support/Space_Engineering_Technology/SpaceWire-AMBA_-_HDL.

[9] L. Sterpaio, A. Marino, P. Nannipieri, G. Dinelli, D. Davalle, and L. Fanucci, "A complete egse solution for the spacewire and spacefibre protocol based on the pxi industry standard," *Sensors*, vol. 19, p. 5013, 11 2019.

[10] ESA. SpW-RMAP-Astrium. https://www.esa.int/Enabling_Support/Space_Engineering_Technology/Microelectronics/SpW-RMAP-Astrium#:~:text=The%20SpaceWire%2DRMAP%20IP%20core,protocols%20in%20FPGAs%20and%20ASICs.

[11] S. Saponara, L. Fanucci, M. Tonarelli, and E. Petri, "Radiation tolerant spacewire router for satellite on-board networking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 22, pp. 3–12, 2007.

[12] T. Takashima, E. Ogawa, K. Asamura, and M. Hikishima, "Design of a mission network system using spacewire for scientific payloads onboard the arase spacecraft," *Earth, Planets and Space*, vol. 70, 12 2018.

[13] A. Leoni, P. Nannipieri, D. Davalle, L. Fanucci, and D. Jameux, "Shine: Simulator for satellite on-board high-speed networks featuring spacefibre and spacewire protocols," *Aerospace*, vol. 6, 2019.

[14] ECSS. ECSS-E-ST-50-52C – SpaceWire – Remote memory access protocol. https://ecss.nl/standard/ecss-e-st-50-52c-spacewire-remote-memory-access-protocol-5-february-2010/.

[15] AMD Xilinx. Xilinx Zynq-7000 SoC ZC702 Evaluation Kit. https://www.xilinx.com/products/boards-and-kits/ek-z7-zc702-g.html.

[16] Microchip. PolarFire FPGA Evaluation Kit. https://www.microsemi.com/existing-parts/parts/138273.

[17] AMD Xilinx. Zynq-7000 All Programmable SoC Verification IP v1.0. https://www.xilinx.com/support/documentation/ip_documentation/processing_system7_vip/v1_0/ds940-zynq-vip.pdf.

[18] Microchip. ProASIC3. https://www.microsemi.com/product-directory/fpgas/1690-proasic3.

A hybrid and reconfigurable edge node computer using SpaceWire and SpaceFibre

Shinpei Kondo
Space Engineering Division
NEC Space Technologies, Ltd.
Fuchu, Tokyo, Japan
s-kondou-ju@nec.com

Hiroki Hihara
Space Engineering Division
NEC Space Technologies, Ltd.
Fuchu, Tokyo, Japan
hihara@nec.com

Mitsuhisa Yamaji
Space Engineering Division
NEC Space Technologies, Ltd.
Fuchu, Tokyo, Japan
m-yamaji@nec.com

Fumio Hodoshima
Shimafuji Electric Incorporated.
Ota-ku, Tokyo, Japan
hodo@shimafuji.co.jp

Takeshi Inuo
Shimafuji Electric Incorporated.
Ota-ku, Tokyo, Japan
inuo@shimafuji.co.jp

Kuniyuki Omagari
Space Systems Division,
NEC Corporation
Fuchua, Tokyo, Japan
k-omagari@nec.com

Takeshi Takashima
Institute of Space and Astronautical Science (ISAS),
Japan Aerospace Exploration Agency (JAXA)
Sagamihara, Kanagawa, Japan
takashima.takeshi@jaxa.jp

***Abstract*— We developed a hybrid and reconfigurable edge node computer with SpaceWire and SpaceFibre interfaces, which is called Space Cube® mk4, as a digital development platform for a model-based development process. Dynamically reconfigurable processors (DRPs) are integrated with conventional microprocessors and a field programmable gate array (FPGA). In addition to artificial intelligence processing capabilities of DRPs, interfaces for general purpose graphics processing units (GPGPUs) and vector processing units of super computers are also incorporated. The processing signals generated by DRPs are transferred in wire-rate via SpaceWire and SpaceFibre interfaces. A breadboard model (BBM) of Space Cube® mk4 weights under 500 g, and a satellite system simulator is under development using the BBM. We report the development result of the BBM in this paper.**

Keywords*— *Dynamically Reconfigurable Processor (DRP), SpaceWire, SpaceFibre, real-time operating system, GPGPU, artificial intelligence

I. Introduction

Reducing the development duration is often required while performing verification tests from various point of views in the development process of onboard digital equipment of satellites. We have developed Space Cube® mk4 to realize a model-based development process as one solution to meet the requirement. Space Cube® mk4 uses three types of processing elements (PEs) to provide a system level simulator for the model-based development process. They are conventional micro-processor units (MPUs), a Field Programmable Gate Array (FPGA), and two Dynamically Reconfigurable Processors (DRPs) [1].

A DRP is a kind of many core processor, and it has high-speed context switching capability. It has several context planes in a chip, and it can realize context switching even in every clock cycle. It enables wire-rate processing speed on up to 10 MHz signal inputs, and it is suitable for matrix operations, filter functions, and neural network emulations. In other words, high-speed processing functions performed by conventional hardware circuitry can be realized by programmable software operations. We apply Space Cube® mk4 for the development of Destiny$^+$ satellite system. A DRP is IP (Intellectual Property) core integrated with a conventional microprocessor, and it extends wire-rate signal processing flexibility of a central processing unit (CPU). Arbitrary digital signals at least 10 MHz can be generated by C-language programs by using a DRP. That enables arbitrary signal transmission over SpaceWire ports by sufficiently low power consumption for onboard computers. Artificial intelligence (AI) related convolutional calculations are performed by a DRP in low power consumption. As for over 1 GHz signal transmissions via SpaceFibre interfaces, we use an FPGA for signal processing. Higher level signal processing and protocol handlings are performed by software on conventional micro-processors mounted on a Space Cube® mk4. We implemented our original router IP on a FPGA with SpaceWire and SpaceFiber interfaces. The combination of DRPs, an FPGA, and conventional processors exploits wide range transmission rates of an integrated SpaceWire and SpaceFibre router.

Extensive simulations will be performed by a hardware-in-the-loop simulator (HILS) during the development phase of the subsystems of satellites. Model-based development process using a Space Cube® mk4 as the HILS is employed prior to system level integration tests to follow the tight development schedule. Hybrid and reconfigurable computing technologies are exploited on the ground-based model to pursue digital development process of newly developed equipment. Dynamically reconfigurable devices are used as the central processing unit for system level simulations and integration tests.

II. DESTINY+ MISSION

A. Mission objectives

DESTINY+ is an engineering and science mission program with distinct but complementary objectives in both disciplines [2, 3]. The disciplines are low-cost and high-frequency deep space exploration utilizing small launch vehicles and high-performance deep space probe platforms. In line with the JAXA's vision, DESTINY+ will demonstrate several necessary space technologies. Primary technology demonstration objectives include 1) to develop spaceflight technologies using electric propulsion and expand the range of its utilization, and 2) to expand the opportunities for small body exploration by acquiring advanced asteroid flyby exploration technologies.

B. Space Craft Overview

DESTINY+ (Demonstration and Experiment of Space Technology for INterplanetary voYage with Phaethon fLyby and dUst Science) is a 480 kg spacecraft with 60 kg of Xe propellant, capable of providing 4 km/s ΔV by IES. Fig. 1 shows its overview. The dry mass and the propellant mass are defined by multi-objective optimization, taking into account the launch capability of the Epsilon S launch vehicle [4, 5, 6]. The spacecraft is equipped with three scientific instruments to observe the surface properties of Phaethon and other potential targets. They will also observe the physical and chemical properties of the dust they produce. Two onboard cameras, a Telescopic Camera for Phaethon (TCAP) and a Multi-band Camera for Phaethon (MCAP), will perform optical observations during flybys. The former uses a single-axis motor to drive the telescope mirror to follow the target, whereas the latter has no moving parts and has a wide field of view. TCAP is also used 2 as an optical navigation sensor to improve the accuracy of orbit determination during the flyby. The DESTINY+ Dust Analyzer (DDA) is developed by the University of Stuttgart. It counts the dust collected along with its impact state [7]. DESTINY+ will be launched in Fy2024.

Fig. 1. DESTINY+ spacecraft overview https://jda.jaxa.jp/result.php?lang=j&id=e0e753bfc25f746565618b16d8522fb2

III. GROUND SUPPORT EQUIPMENT

A. Hybrid and reconfigurable computing

Three types of processing elements (PEs) are used to provide a system level simulator for the model-based development process of DESTINY+. They are conventional micro-processors, a Field Programmable Gate Array (FPGA), and Dynamically Reconfigurable Processors (DRPs). They are integrated on a Space Cube® mk4. It has a LEON5 processor, which was developed by COBHAM/Gaisler, and it is used in combination with a Xilinx Kintex UltraScale FPGA. Fig. 2 shows the block diagram of Space Cube® mk4.

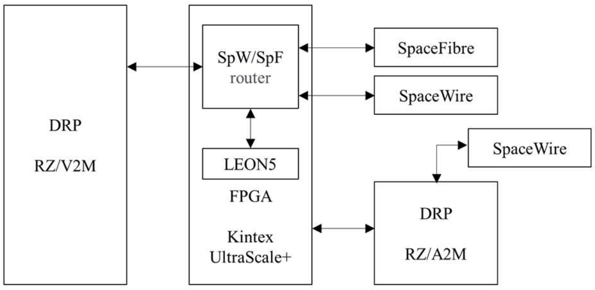


Fig. 2. Space Cube® mk4 block diagram

Wire rate processing is carried out by an FPGA, and the other functions are performed by a conventional micro-processor LEON5. LEON5 is provided as an IP (Intellectual Property), and it can be embedded in the FPGA. Timeliness operations are performed using the combination of a LEON5 processor and FPGA circuitries. Space Cube® mk4 also has two micro-controllers, an RZ/V2M and an RZ/A2M, made by Renesas Electronics [1]. Each micro-controller has a Dynamically Reconfigurable Processor (DRP). DRPs realize very low power consumption. We implemented a SpaceWire and SpaceFibre router on a Xilinx Kintex UltraScale FPGA. It has three SpaceWire ports and two SpaceFibre ports. T-Kernel real-time operating systems is running on the LEON5 processor to control the SpaceWire/SpaceFibre router.

Space Cube® mk4 is used as a subsystem and a system level simulator as an onboard digital equipment subsystem of DESTINY+. An Space Cube® mk4 is planned to be used as a HILS of the verification target of the simulator. Space Cube® mk4 has been developed as a HILS, and it aims at demonstrating hybrid and reconfigurable computing technology for mission data processors. Fig. 3 shows the outlook of Space Cube® mk4 and its interior assembly. The first verification target is a high-speed mission data transmission between sensors and a data recorder. SpaceWire and SpaceFibre packet transmissions are performed by the FPGA.

B. High speed interface for scalability

Giga-bit signal transmission is expected for onboard processing equipment as well as ground support equipment especially for mission data processing. In addition to that, environment and orbit model must be simulated by a simulator constructed with ground support equipment. We are planning to use a vector processor card with tera-flops operation capability for the simulation of the environment and orbit model simulations. Giga-bit signal transmission interfaces are required for these purposes. In consequence, we provide 2.5 Gbps GEN3 interfaces of PCI Express on a Space Cube® mk4 for 10 Gbps multi-lane transmission. Since the Space Cube® mk4 aims at a prototyping of an onboard mission data processor on a satellite, two SpaceFibre interface channels are accommodated for Giga-bit signal transmission in addition to conventional SpaceWire interface ports. As for the vector processor applications, we use PCI Express interfaces in the early phase of the development. We see the porting of applications from PCI Express interfaces to SpaceFibre interfaces is straightforward.

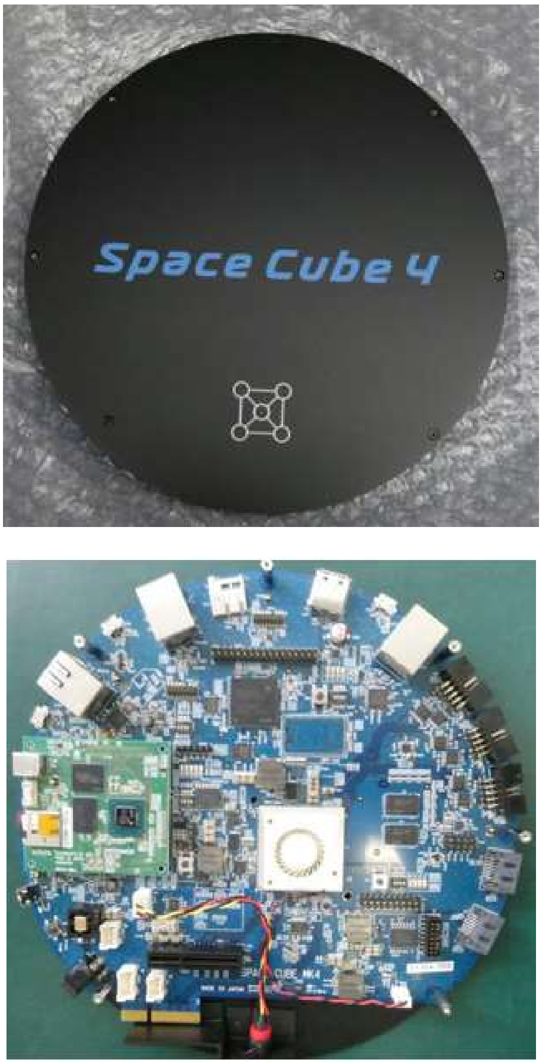


Fig. 3. The outlook of Space Cube® mk4 and its interior assembly

IV. MODEL-BASED DEVELOPMENT

A. *Space Cube ®mk4 architecture*

Higher level signal processing and protocol handlings are performed by software on conventional micro-processors mounted on a Space Cube® mk4. It is practical way to establish design framework as a software development kit (SDK) to make developers understand the reference model of higher level onboard processing. The reference model includes onboard computer architecture, communication model, and database scheme for satellite operations. Space Cube architecture [8] is a reference for Space Cube® mk4, which defines the required specification of general-purpose onboard computers. Several satellites like the earth observation satellites, LEO scientific satellites and inter-planetary scientific satellites have been investigated to establish the design framework. Based on the assessment, standard middle ware requirement, telemetry/command design criteria, and network design criteria have been published. The architecture features following requirements.

1) Space Cube mk4 Architecture was derived from T-Engine architecture. T-Engine is an open platform for embedded use, which is applicable for various kinds of microprocessors.

2) SpaceWire and SpaceFibre are mandatory interfaces for realizing scalable network based on spacecraft architecture.

3) Compatibility is maintained through the standard middleware specification and API to accommodate various types of processing elements.

4) To satisfy small size, light weight, low power consumption and low cost requirement for small satellites, an embedded microcontroller within an FPGA and peripheral I/O channels embedded in itself are recommended.

B. *DRPs in a Space Cube mk4*

There are two types of DRPs in a SpC mk4. One DRP, which is called RZ/A2M, is used as a small-scale embedded controller with an Arm®-based micro-controller. It has several input/output (I/O) circuitries. This type of DRP can create arbitrary signals, and up to 10 MHz bit stream transmission with arbitrary wave forms can be achieved with fully software control. Thanks to this high-speed processing capability, some kinds of noise figures can be simulated on the transmission lines of inputs and outputs. The other one, which is called RZ/V2M, is a large scale DRP with a high-performance Arm®-based microprocessor. Heavy load processing as neural network functions for artificial intelligence applications can be implemented with sufficient low power consumption as an edge node processor. By utilizing these processors, it is possible to transferring the built-in software efficiently in short time, and the switchover from a simulation to the actual device evaluation becomes smooth.

C. *Integration of the DRPs and SpaceWire/SpaceFibre interfaces*

RZ/V2M processor is connected to FPGA through PCI Express interfaces, and the signals generated by RZ/V2M can be transmitted at high speed to the onboard equipment from the SpaceWire and SpaceFibre interfaces mounted on the FPGA. RZ/A2M processor has the SpaceWire interface, and the signals generated by RZ/A2M processor, for example random signals, can be transmitted at 10MHz to the onboard equipment. The standard middleware is based on Space Monitor & Control Protocol (SMCP). This protocol is performed by a conventional MPU. SMCP was developed by JAXA/ISAS [9, 10]. The protocol aims at unified building method of commands, telemetry messages, and sequence for all satellites and onboard equipment. Telemetry and Command processing functions are realized through SMCP. Reliability and timeliness were taken into account by exploiting Remote Memory Access Protocol (RMAP) and Time-Code delivery function. Retry and Redundancy control are carried out using Cyclic Redundancy Check (CRC) in RMAP packet. Quality of Service (QoS) functions are provided by SpaceFiber protocol. Scheduling (Slot Control) are implied by Time-Code. Fig. 4 shows the application development scheme.

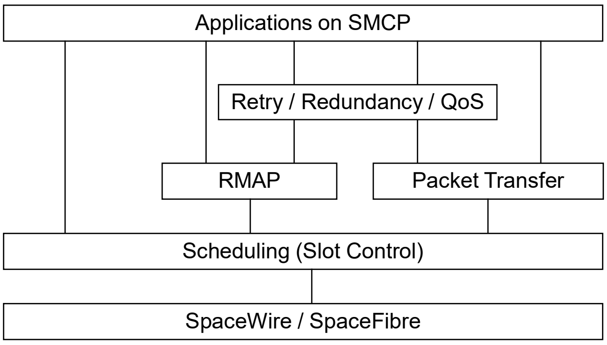


Fig. 4. Application development scheme using, DRPs and SpaceWire/SpaceFibre interfaces

V. CONCLUTION

We have developed Space Cube® mk4 as a versatile simulator used as ground support equipment as well as a prototype of an onboard mission data processing computer. It accommodates three types of processing element, a conventional micro-processor, an FPGA, and a dynamically reconfigurable processor (DRP). The method to exploit hybrid and reconfigurable computing technology has been established based on Spacecraft Monitor & Control Protocol (SMCP) designed by JAXA/ISAS. Model-based development process is employed prior to system level integration test to follow the tight development schedule. Dynamically reconfigurable devices and a SpaceWire/SpaceFibre router are used as key components, and extensive simulation is performed by a hardware-in-the-loop simulator.

REFERENCES

[1] https://www.renesas.com/jp/en/products/microcontrollers-microprocessors/rz-cortex-a-mpus/rzv2m-dual-cortex-a53-lpddr4x32bit-ai-accelerator-isp-4k-video-codec-4k-camera-input-fhd-display-output

[2] H. Toyota, K. Nishiyama, Y. Kawakatsu, Y. Sato, T. Yamamoto, S. Okazaki, T. Nakamura, R. Funase, T. Inamori, T. Arai, K. Ishibashi, M. Kobayashi, D. Team, “Destiny+: Deep space exploration technology demonstrator and explorer to asteroid 3200 phaethon, in: Low-Cost,” Planetary Missions Conference, 2017.

[3] N. Ozaki, T. Yamamoto, T. F. Gonzalez, R. R. Gutierrez, N. Pushparaj, T. Chikazawa, D. A. Dei Tose, O. C elik, N. Marmo, Y. Kawakatsui, T. Arai, K. Nishiyama T. Takashima, " Mission Design of DESTINY+: Toward Active Asteroid (3200) Phaethon and Multiple Small Bodies," Acta Astronautica, arXiv:2201.01933v1, 6 Jan. 2022.

[4] F. Zuiani, Y. Kawakatsu, M. Vasile, “Multi-objective optimisation of many-revolution, low-thrust orbit raising for destiny mission, in: Advances in the Astronautical Sciences,” Vol. 148, 2013, pp. 783–802.

[5] C. H. Yam, F. Zuiani, Y. Kawakatsu, T. Yamamoto, M. Vasile, “Orbit raising trajectory and system analysis for the mission destiny,”, in: 24th International Symposium on Space Flight Dynamics, Laurel, Maryland, US, 2014.

[6] T. Yamamoto, C. H. Yam, S. Campagnola, Y. Sugimoto, A. Oyama, T. Tatsukawa, C. Hirose, T. Ikenaga, Y. Kawakatsu, S. Ogura, S. Sato, D. Team, “Destiny trajectory design, in: 30th nternational Symposium on Space Technology and Science,” Kobe, Japan, 2015.

[7] H. Krüger, P. Strub, R. Srama, M. Kobayashi, T. A. Arai, H. Kimura, T. Hirai, K. G. Moragas, N. Altobelli, V. J. Sterken, J. Agarwal, M. Sommer, E. Grün, “Modelling DESTINY+ interplanetary and interstellar dust measurements en route to the active asteroid (3200) Phaethon,” Planetary and Space Science 172 (2019) 22–42. doi:10.1016/j.pss.2019.04.005.

[8] T. Takahashi, T. Takashima, S. Kuboyama, M. Nomachi, Y. Kasaba, T. Tohma, H. Hihara, S. Moriyama, T. Tamura, "SpaceCube 2 -- An Onboard Computer Based on SpaceCube Architecture", International SpaceWire Conference 2007, 17-19 September 2007, p.65-68.

[9] T. Yamada, "Functional Model of Spacecraft (FMS)", GSTOS 201, 15 September 2009.

[10] T. Yamada, "Spacecraft Monitor & Control Protocol (SMCP)", GSTOS 200, 15 September 2009.

Application of SpaceWire and SpaceFibre in GR765

Joaquin Espana Navarro
Cobham Gaisler AB
Göteborg, Sweden
joaquin@gaisler.com

Fabio Malatesta
Cobham Gaisler AB
Göteborg, Sweden
fabio.malatesta@gaisler.com

Jan Andersson
Cobham Gaisler AB
Göteborg, Sweden
jan@gaisler.com

Roland Weigand
ESA - ESTEC
Noordwijk, Netherlands
roland.weigand@esa.int

Abstract— **The GR765 is a radiation-hardened system-on-chip planned to be the successor of the GR740 quad-core LEON4FT processor. The GR740 LEON4FT quad-core processor is the highest performing LEON-based component currently available for space applications and is being applied in various missions and spacecraft architectures.**

The GR765 architecture includes several improvements over the GR740, most notably the addition of the bootstrap option to select between the LEON5FT and NOEL-V FT high-performance processors that will further increase computational performance over the LEON4FT used in the GR740. The GR765 provides a low-threshold upgrade path for current GR740 users that need additional computational performance, improved power performance, or that would benefit from the extended functionality in the new architecture.

Keywords— ***SpaceWire, SpaceFibre, SoC, fault-tolerant, RISC-V, NOEL-V, LEON, SPARC***

I. Introduction

The LEON line of processors has a large user base within the space industry and, given the two decades of use of the LEON SPARC instruction set architecture (ISA), there is a significant amount of software heritage that can be reused. The LEON5 provides backward compatibility through implementation of the same SPARC V8(e) instruction set, adds new features to address mixed-criticality and safety-critical systems and improves computational performance.

Several generations of LEON processors have included SpaceWire support, including the UT699 and UT700 LEON3FT processors and the GR712RC dual-core LEON3FT processor. The GR740 introduced an on-chip SpaceWire router, allowing users to transition from architectures that required a microprocessor and a separate router IC. The GR765 continues this trend of higher integration by increasing the number of SpaceWire ports in the on-chip router and adding support for both SpaceFibre and WizardLink.

The RISC-V ISA has gained momentum in other industries and represents a modern alternative with the same openness as was the case for SPARC. The ISA has recently been selected for the US HPSC development [1]. The prospect of aligning on one open architecture in both Europe and the US is interesting due to the synergy effects that can be attained where development projects in both EU and US may accelerate one another.

II. Architecture

The block diagram of the GR765 microprocessor is depicted in Fig. 1. The GR765 features set has several improvements compared to the GR740. The number of processor cores is increased from four to eight, the LEON5FT processor is employed instead of the LEON4FT and a bootstrap option allows to select eight NOEL-V RISC-V cores instead of eight LEON5FT. The LEON5FT microprocessor is a superscalar implementation that combined with improved

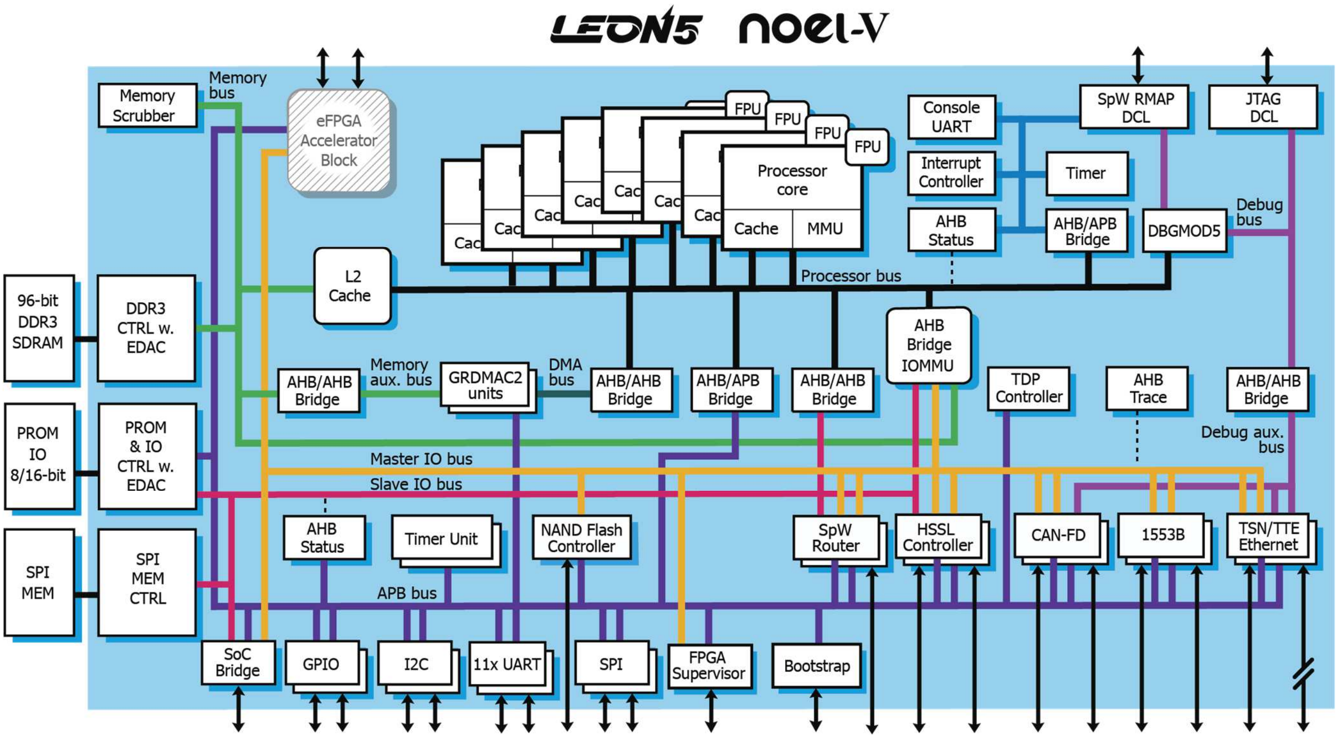


Fig. 1. GR765 block diagram

branch prediction capabilities offer a significant performance improvement over the LEON4FT at the same processing frequency. The NOEL-V FT is a superscalar RISC-V processor with performances comparable to the LEON5FT. The processor core frequency target is also increased compared to the GR740: from 250 MHz to 1 GHz.

Doing more work per clock cycle puts additional pressure on the on-chip interconnect and on main memory systems. The GR765 will have an upgraded interconnect between the processor cores and the Level-2 cache, providing multiple parallel data paths. The primary external memory interface is currently specified as DDR2 and DDR3 SDRAM with dual x8 device correct capability.

Other improvements include new dedicated DMA controllers and other interfaces such as I2C, additional SPI, MIL-STD-1553B and NAND Flash.

The target technology for the implementation is STMicroelectronics 28nm FDSOI.

III. SpaceWire

The SpaceWire router is implemented as defined in the ECSS-E-ST-50-12C standard. The GR740 has a SpaceWire router with eight external and four internal ports. The GR765 extends this to have twelve external ports. The external SpaceWire ports are complemented by the on-chip AMBA ports, allowing the distribution of AMBA traffic to and from the SpaceWire network. Each AMBA port has also an RMAP target and several Direct Memory Access (DMA) channels. The router can be used for two main purposes:

- Provide SpaceWire connectivity to the processor cores.
- Routing capabilities for external SpaceWire nodes.

It is important to underline that the protection mechanisms implemented in the GR765 also allow the SpaceWire router to act completely separately from the rest of the processor and transparently to the software. An RMAP target provides access to the router’s configuration port and therefore the router can be configured by external SpaceWire nodes.

IV. High-speed serial links

The GR765 instantiates the GRHSSL IP in order to support both SpaceFibre and WizardLink protocols over the same physical link. Four external links are currently specified to be implemented. The GR765 supports multiple bit rates as defined in the SpaceFibre standard, including 5 and 6.25 Gbps.

GRHSSL provides an interface between the AHB bus and the high-speed serial link. It implements both SpaceFibre and WizardLink controllers. The SpaceFibre controller complies with the SpaceFibre specification ECSS-E-ST-50-11C, whereas the WizardLink codec has been designed to inter-operate with the TLK2711 SerDes transceiver from Texas Instrument, although the IP can interface other SerDes devices, either on- or off-chip.

Fig. 2 shows the block diagram of the GRHSSL IP. The core features a flexible DMA layer with a configurable number of DMA channels, each one with its own AHB master interface to fetch and store descriptors and data from/to external memory. The number of DMA channels per GRHSSL instantiation is not defined yet at the time of writing this document.

The IP is configured through registers accessed through an AHB slave interface. SpaceFibre and WizardLink have separate sets of AHB registers, meaning that the IP will include two different AHB I/O banks.

Only one of the two controllers can be active at a time, which is selectable at run time via the AHB registers. The active controller will communicate with the AHB bus and drive the SerDes interface.

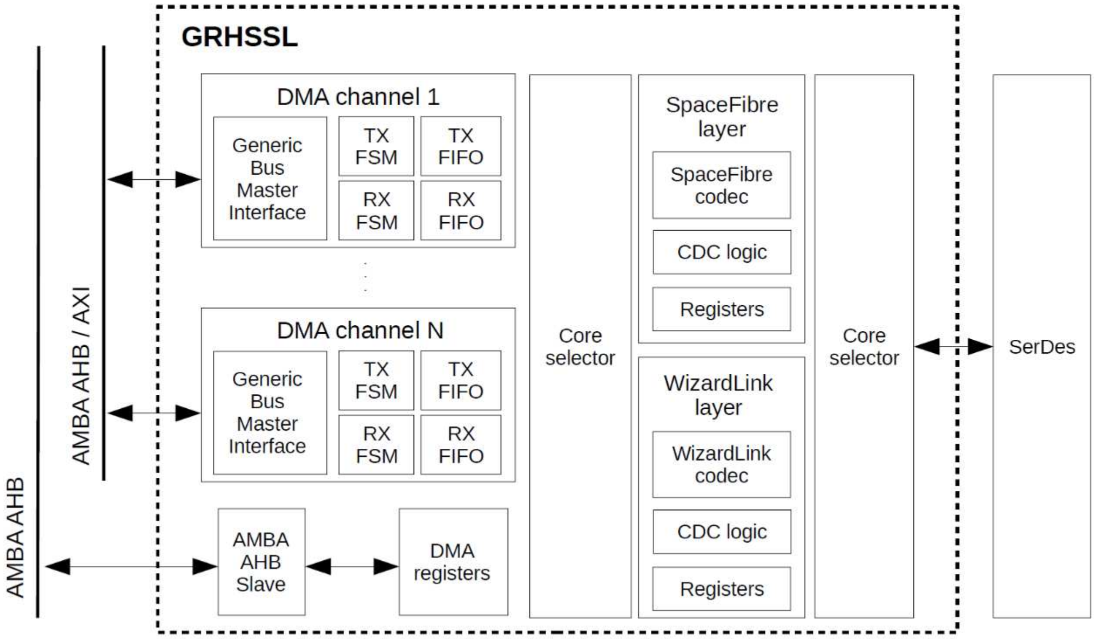


Fig. 2. GRHSSL IP block diagram

A. *SpaceFibre support*

The SpaceFibre codec has been designed in accordance with the SpaceFibre specification ECSS-E-ST-50-11C. The codec is composed of several layers, each one covering a specific part of the SpaceFibre functionality, namely: Interface Layer, Lane Layer, Retry Layer, Virtual Channel Layer and Broadcast Channel Layer.

The codec is served by the DMA layer in order to transmit and receive packets. In runtime, the user shall assign the Virtual Channels (VC) and the Broadcast Channel (BC) to the existing DMA channels. There are no restrictions on how many VC/BC channels can be handled by each DMA channel: arbitration among the channels is resolved in a Round-Robin fashion, except for the Broadcast Channel which always has the highest priority.

The transmission and reception of packets over a VC is controlled via hardware descriptors that determine where the packet shall be fetched from or stored to. The user is in charge of adding descriptors to external buffers and inform the IP that there are new descriptors available to fetch. When idle, the DMA channels check which channels have descriptors available and data to process. Once a channel is selected, the FSM in the DMA channel will fetch the descriptor and handle the data accordingly. The process is completed once the DMA engine disables the descriptor by overwriting its content in memory. For the BC, the process does not require descriptors; instead, the data is written to a circular buffer handled via hardware pointers.

The IP can generate interrupts whenever a message has been transmitted or received, as well as when specifics events are detected such as link errors. The conditions triggering interrupts are separately configurable via the AHB registers.

B. *WizardLink support*

GRHSSL features a companion WizardLink controller initially designed to inter-operate the TLK2711 transceivers, although the IP can interface other SerDes devices. Due to the open nature of the WizardLink communication, the functionality included in the controller is minimal, and most of the parameters are configurable so that the IP can interface any equipment implementing its own custom protocol over a high-speed serial link. Most of the functionality of the codec correspond to the Interface Layer of the SpaceFibre controller, namely symbol synchronization, 8b10b encoding and clock-domain-crossing techniques.

When operating in WizardLink mode only the first DMA channel is used. As with SpaceFibre, the transmission and reception of packets are controlled by hardware descriptors which are added by the user. However, not only data characters but also control characters need to be written and read from external memory, since the protocol does not specify the meaning of the control characters. That implies that the descriptors have to be extended to indicate the addresses to both the data and K-flags of the packet. It is therefore the responsibility of upper layers (i.e., software) to establish the functionality linked to the control characters.

In order to enhance protocol support in hardware, the DMA engine includes programmable commands that allow the user to transmit and receive control words without needing to set up descriptors. The core can generate interrupts when these commands are transmitted or received. This is an optional feature that can be used to easily insert and extract control words that do not necessarily belong to a WizardLink packet, for instance when starting or stopping the link, or as part of a custom flow control mechanism. The core currently supports up to 8 programmable commands which are controlled via AHB registers.

C. *Planned extensions*

As part of the GR765 development, there are two major extensions planned for the GRHSSL IP: RMAP support and integration with the SpaceWire router. They are described in the paragraphs below.

RMAP support conforming with the ECSS-E-ST-50-52C specification will be added to the DMA engine of the SpaceFibre layer of GRHSSL. The same standard as for SpaceWire can be used due to the fact that SpaceFibre and SpaceWire are compatible at the network layer. Each DMA channel will feature its own RMAP target to make full utilization of the dedicated AHB bus that is available to every DMA channel.

The user will have the option to enable RMAP decoding separately for each Virtual Channel at run time. When receiving a packet over a given VC, the DMA channel will check if it is an RMAP packet provided that RMAP decoding is enabled for the VC in question. If so, the RMAP target autonomously handles the access to the AMBA space of the device, otherwise the packet is processed by the DMA channel using hardware descriptors.

The second addition is the integration of GRHSSL in the existing SpaceWire router to provide a bridge between SpaceFibre and SpaceWire traffic. GRHSSL will have a number of external FIFO ports to directly expose Virtual Channel data. This number is determined at implementation time by means of a VHDL generic. The user can then configure at run time via the AHB registers which VCs are set in bypass mode, i.e., connected to the external FIFO interfaces and thus not handled by the DMA engine. This also implies that any potential RMAP command received over a VC in bypass mode is not handled by the RMAP target of the IP but passed on to the external ports transparently. Finally, the external interfaces of GRHSSL will be connected to FIFO ports of the SpaceWire router to provide the required bridging capabilities.

Work is also ongoing to guarantee the interoperability between the GRHSSL IP and the STMicroelectronics 28nm FDSOI SerDes macro.

V. Schedule and status

The GR765 is planned to enter manufacturing of prototypes in 2023. The presentation will describe the overall component development as well as the current status of implementation of the SpaceWire and SpaceFibre protocols.

References

[1] Microchip to develop next generation 12 core RISC-V space processor for NASA, 2022-08-16 [online] https://www.eenewseurope.com/en/microchip-to-develop-next-generation-space-processor-for-nasa/

RC64: Space Manycore Enabled by SpaceWire & SpaceFibre

Ran Ginosar
Ramon.Space, Ltd.
Yoqneam Illit, Israel
ran.ginosar@ramon.Space

Peleg Aviely
Ramon.Space, Ltd.
Yoqneam Illit, Israel
peleg.aviely@ramon.Space

***Abstract*—RC64 rad-hard manycore DSP/ML processor employs the Space-oriented SpFi and SpW communication protocols. It is the only known Space product fully based on these two standards. RC64 employs six SpaceWire links and twelve high-speed SpaceFibre links. This connectivity facilitates modular complex systems comprising tens and more processors spanning multiple PCBs and multiple enclosures for all Space missions. Small and reliable, thermal-cycle resilient chip packages are enabled. Systems for satellite communications, Earth observation, remote sensing, storage, cloud computing, autonomous spacecraft and instruments and more are being designed and implemented. Design considerations and challenges are described, and future versions are described.**

Keywords—SpaceWire, SpaceFibre, Manycore, DSP, Rad-Hard

I. RC64 Architecture

A. Processing

RC64 integrates 64 DSP cores, 4 Mbyte of shared memory, a HW scheduler, accelerators and peripherals [1]. It is designed for effective and power-efficient high-performance Space applications [2]. It has been fabricated in 2017 (65nm TSMC, see Fig. 1) and is qualified to MIL-PRF-38535 Class Level S (300 kRad, effectively no SEL, low SEU). It includes massive HW facilities for FDIR and *Virtual Radiation Shield*.

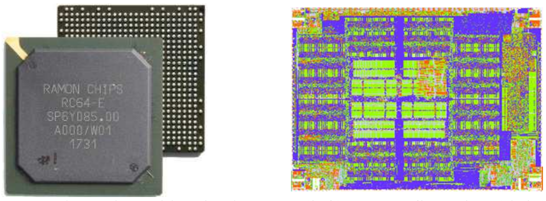


Fig. 1. RC64 is packaged in plastic BGA (left). RC64 floorplan (right) shows a large shared memory block surrounded by 64 DSP/ML cores.

B. SpaceWire

Two SpaceWire (SpW) links include RMAP capabilities and enable full control by an external host processor.

Four SpW links connect via DMA to the shared memory, facilitating packet networking. Store-and-forward routing in RC64 is software-based, complying with the SpaceWire standard [3]. RMAP extension complies with [4].

C. SpaceFibre

Twelve high-speed serial links are included in RC64. All of them can operate as SpaceFibre (SpFi). Some of the links can double as SRIO interfaces. The transmit and receive parts of each link may connect to different targets; for example, 12 SpFi inputs may arrive from chip X while 12 SpFi outputs are feeding into chip Y, facilitating stream processing (see below).

SpFi links are managed by DMA. They read from (and write to) shared memory. Sending and receiving SpFi data are separated in time from any processing of the data by the DSP cores. SW is used to process the data as well as to route SpFi packets and perform other Network Processor functions. Similar to SpW, SpFi store-and-forward routing in RC64 is software-based, complying with the SpaceFibre standard [5]. To facilitate flexible and fault-tolerant routing, logical address routing is employed, rather than path addressing. Routing software also facilitates a bridge between SpFi and SpW networks, allowing messages to cross over, when feasible due to size limits and QoS concerns. Conceptual implementation of both SpW and SpFi routing on RC64 is depicted in Fig. 2

II. SpW Implementation on RC64

Four SpW links on RC64, which do not support RMAP, are implemented using LVDS ports, enabling long-range and board-to-board connectivity. The two SpW ports that do facilitate RMAP employ LVCMOS, saving I/O pins when intra-board connections are needed. In principle, SpW links are connected to a HW switch that detects RMAP accesses (arriving on either one of the two the RMAP-enabled links) by means of their logical address and switches them to an RMAP unit. All other ingress SpW packets are switched over to the SW router: the ingress packets are stored onto shared memory, and the SW router is notified. This conceptual internal structure, in support of SpW networking, is shown in Fig. 2.

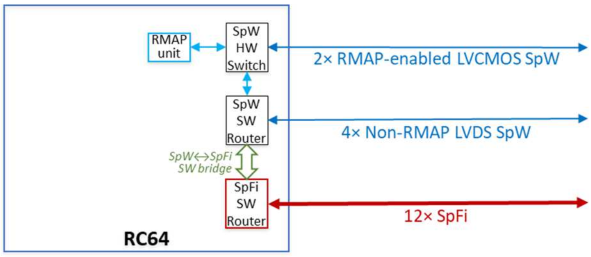


Fig. 2. Conceptual SpW switch (HW) and router (SW), SpFi router (SW) and SpW-to-SpFi software bridge in RC64

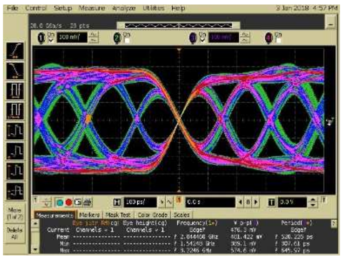
Radiation tests of SpW links have shown high resilience to all radiation effects.

III. SpFi Implementation on RC64

Ramon.Space hardened the ‘hard IP core’ SERDES ports to enhances its SEL tolerance. SEL tests demonstrated a high SEL threshold, suitable for LEO, GEO and Deep Space missions.

The SERDES ports and associated ‘MAC layer’ logic enable flexible data rate, controllable only by SW.

Eye diagram and data rate tests were conducted, showing high performance up to the full 5 Gbps data rate at a wide range of cable distances. An example eye diagram of RC64 SpFi port operating at 6.25 Gbps is shown in Fig. 3.


File Control Setup Measure Analyze Utilities Help
3 Jan 2018 4:57 PM
100 mV/
100 ps/
0.0 s
0.0 V
Measurements Markers Mask Test Color Grade Scales

```
            Eye width(cg)  Eye height(cg)  Frequency(1+)   V p-p( )     Period( +)
Current     Channels < 1   Channels < 1    Edge?           476.3 mV     Edge?
  Mean      -------------  -------------   ? 2.044408 GHz  481.422 mV   ? 520.225 ps
  Min       -------------  -------------   ? 1.54249 GHz   389.1 mV     ? 307.61 ps
  Max       -------------  -------------   ? 3.2246 GHz    574.6 mV     ? 645.97 ps
```

Fig. 3. 6.25 Gbps SpFi Eye Diagram

SpFi implementation on RC64 enables two virtual channels. The HW covers the physical, lane and data link layers; the multi-lane layer is not implemented. The lane and data link layers allow separating incoming and outgoing parts of the link, creating unidirectional links.

SpFi routing on RC64 is based on software, similarly to SpW routing, as indicated in Fig. 2. The two SW routers are optionally bridged within RC64, enabling packets to traverse mixed protocol sub-networks.

IV. RC64-Based System with and Without a Host

A host, implemented as a CPU such as Ramon.Space GR712RC or an FPGA such as Microchip PolarFire, can be connected to the two RMAP-based ports. In fact, a ring of RC64 chips can be connected to one host, enabling resilience to any single fault. Any RC64 may boot its SW directly from an attached boot Flash. Alternatively, it can be booted up from another RC64 chip or FPGA, through RMAP SpW. Fig. 4 demonstrates such a system.

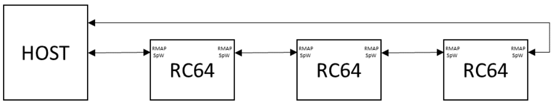


Fig. 4. A ring of RC64 devices is controlled by a host CPU through RMAP SpW links. Each RC64 can booth either from its local boot flash or from the host.

One example of a system without a host is VPX64, a single board computer based on RC64 (Fig. 5). It only boots from storage. Another example is NuStream, where RC64 may boot either from boot Flash or from a remote host, bridged through an attached FPGA (Fig. 6).

Fig. 5. VPX64 board (3U-VPX form factor) showing RC64. Boot flash is mounted on the back side of the board. No Host is used.

Fig. 6. NuStream board: Front side (left) carries RC64. The FPGA is shown on the back side (right).

V. RC64-Based System Using FPGA Acceleration

Ramon.Space single-card compute & storage product NuStream (Fig. 6) employs a Microchip PolarFire FPGA directly connected to RC64 via two SpW and four SpFi links. The SpW and SpFi networking scheme on NuStream is detailed in Fig. 7.

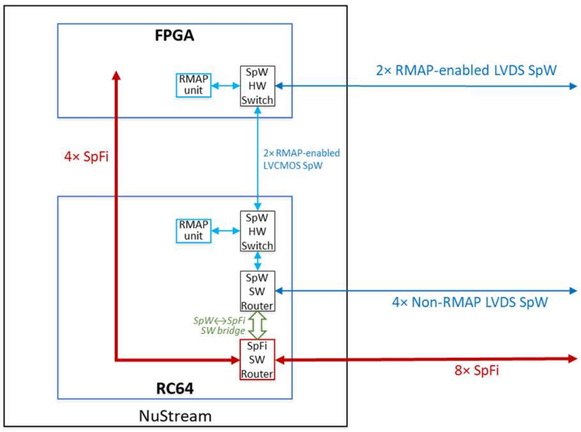


Fig. 7. SpW and SpFi networking in NuStream.

The FPGA on NuStream may execute a variety of tasks such as high performance EDAC for attached 3D NAND flash units for a 1TB Space quality storage drive. In another example, the attached FPGA performs encryption and decryption, bit-level tasks that are more suitable for gate-level FPGA logic than for SW-based RC64. In yet another example, the FPGA is programmed as interface to enable connecting NuStream to non-SpW, non-SpFi devices such as cameras using Camera Link, PCIe links, and Ethernet links, and bridging between them and the SpFi & SpW network; four software-defined SERDES ports of the FPGA (not show in Fig. 7) are available for implementing such interfaces.

The NuStream board can be reprogrammed to perform Machine Learning tasks on RC64 [6]. The FPGA may be used for off-loading some matrix-multiplication tasks from RC64. The attached DRAM may buffer data, activations, and weights. When needed, a second NuStream board may be accessed over the SpFi & SpW network for streaming storage of large-volume weight files.

SpFi and SpW networking in NuStream demonstrates employing a high level protocol on top of the basic protocols. As shown in Fig. 8, the base packet format comprises a destination address (one byte logical address in our networks), cargo and trailing byte. Higher level protocol packet formats are encapsulated within the basic format: the protocol header and the protocol data are parts in the cargo field. Thus, packet routing is ignorant of any upper layer, or application layer, protocols. For instance, when NuStream is configured as a storage drive, a unique storage oriented application level protocol is employed, carrying storage commands and data, encapsulated in SpFi and SpW packets.

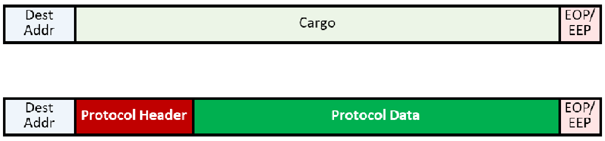


Fig. 8. SpW / SpFi packet format (top), and encapsulated protocol header and data for higher-level protocol packet format (bottom).

Note that the SW router may be programmed to peek into the cargo and provide additional protocol-specific services, if so desired. Note further the exception related to the RMAP-over-SpW protocol: In addition to RMAP-specific packet format [4], RMAP packets are routed and switched based on logical address, to assure their delivery to RMPA units, as detailed in Section II above.

Additional features enabled by the SW routers relate to Quality of Service (QoS) and higher layer services. QoS capabilities include priorities, virtual channels, and bandwidth restrictions. Some of these features follow QoS as specified in [5]. FDIR capabilities include link EDAC and flexible routing tables to compensate for ailing links. End-to-end reliability including retransmission may be implemented at higher level protocols (L4 transport layer, or application); for instance, storage applications on NuStream facilitate end-to-end error checking and recovery. Session layer (L5) capabilities, also implemented in the storage application of NuStream, enable packet sequencing, handling multi-packet messages, tracking and recovery of missing packets, window acknowledging and similar capabilities, all highly useful for implementing reliable storage. Another feature of L4 end-to-end capabilities pertains to packet assembly/disassembly, especially when very long packets (designed with SpFi in mind) should be routed over SpW links, in order not to overflow frame size limitations of the RC64 implementation of SpW. Finally, the L4 transport layer may optionally attach time stamps to packets, in order to facilitate merging of packets that carry related data (such as same-time samples by multiple sensors or antenna elements).

VI. Systems Integrating Many RC64 Processors

The primary goal for using a large number of SpFi and SpW links is to enable a dense network of many RC64 chips, capable of high-performance computations. For instance, a 2D mesh of RC64 units may be connected by channels of three SpFi links in each direction (North, South, East and West). Note that RC64 does not implement a multi-lane protocol—data decomposition to multiple links and the corresponding assembly at the receiving end are done by SW rather than by HW cores.

An alternative interconnect method implemented in RC64 splits the send and receive side of each link so that data can be pipelined over multiple stages of RC64 units, fully utilizing all links. This method is somewhat similar to JESD204B/C links typically employed in ADC and DAC products.

As an example and demonstrator, a regenerative satellite transceiver was constructed where two RC64 units received 'return channel' data and demodulated them using DVB-RCS, while two other RC64 units implemented a DVB-S2 modulator for transmitting the data to a gateway in a telecom demonstration. All four RC64-based boards were VPX64, shown in Fig. 5.

VII. Complex Systems

A large demonstrator of an Edge Compute system is presently being constructed, using multiple extended (220mm) 6U-VPX Edge-Compute cards. Each card carries three NuStream storage and processing boards (Fig. 6) and one MPSoC FPGA-based multicore board for managing the three NuStream boards (Fig. 9). The MPSoC based multicore board is enhanced by means of SpW and SpFi IP cores, configured into the logic part of the MPSoC FPGA. The three NuStream boards and the multicore board are interconnected by several SpW and SpFi links to facilitate redundancy and fault tolerance. In addition, each card is interconnected with all other cards by multiple SpW and multiple SpFi links. Thus, all network nodes and routers form a unified network, enabling any node to access any other node, whether on the same card or on a different card.

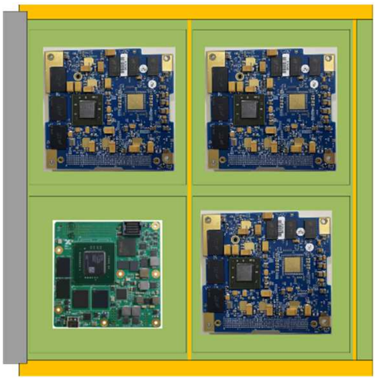


Fig. 9. Edge Compute card (6U VPX 220mm) combining three NuStream boards and one MPSoC multicore board

The result is a very large number of network nodes. Each RC64 implements both SpW and SpFi routers. As noted, the two networks are bridged by SW in each node, so that packets may start, for instance, as SpW packets and end up arriving at their destination as SpFi packets.

On starting up, each board employs its BIST and link-test procedure to note the status of all its incident links. A distributed algorithm gathers all this status information, notes malfunctioning links and nodes, and configures a network that reaches all active nodes with as many redundant paths as possible. Corresponding routing tables are generated for each RC64 node and are distributed over the network. Routing algorithms optionally include group adaptive routing [5] and packet broadcast. Links along redundant paths may be shut down, in order to save power when unused.

VIII. Next Generation Manycore

The future generation of RC64 extends SpFi from 12 links to 32 links, and from 5 Gbps per link to 32 Gbps per link. Some of the links are based on Long Range SERDES enabling board-to-board high speed connectivity while other links use Short Range SERDES, limited to same-board connectivity but dissipating much less power. SpW and SpFi networking are retained the same as in RC64, enabling software compatibility and smooth migration from RC64 based systems to future ones.

At present, SpFi protocol supports only 8B/10B line codes. It is highly desirable to upgrade to more efficient codes, such as 64B/67B (supported by Interlaken) or 128B/130B (supported by PCIe) in order to take better advantage of high speed SERDES links such as 32 Gbps planned for the future generation of RC64.

REFERENCES

[1] Ginosar, R., Aviely, P., Israeli, T., & Meirov, H. (2016, March). RC64: High performance rad-hard manycore. In 2016 IEEE Aerospace Conference (pp. 1-9). IEEE.

[2] Aviely, P., Radovsky, O., & Ginosar, R. (2016, March). DVB-S2 software defined radio modem on the RC64 manycore DSP. In 2016 IEEE Aerospace Conference (pp. 1-10). IEEE.

[3] ECSS-E-ST-50-12C Rev.1 (15 May 2019). SpaceWire—Links, nodes, routers and networks.

[4] EECS-E-ST-50-52C (5 Feb. 2010). SpaceWire—Remote memory access protocol.

[5] ECSS-E-ST-50-11C (15 May 2019). SpaceFibre - Very high-speed serial link.

[6] Ginosar, R. et al. (June 2021). Ramon Space RC64-based AI/ML Inference Engine, in ESA European Workshop on On-Board Data Processing.

Test & Verification (Short)

SpaceWire Test and Development with STAR-System and the SpaceWire PCIe Mk2

Stuart Mills
STAR-Dundee
Dundee, Scotland, UK
stuart.mills@star-dundee.com

Alan Spark
STAR-Dundee
Dundee, Scotland, UK
alan.spark@star-dundee.com

Balint Furdek
STAR-Dundee
Dundee, Scotland, UK
balint.furdek@star-dundee.com

Chris McClements
STAR-Dundee
Dundee, Scotland, UK
chris.mcclements@star-dundee.com

Dave Gibson
STAR-Dundee
Dundee, Scotland, UK
david.gibson@star-dundee.com

Pete Scott
STAR-Dundee
Dundee, Scotland, UK
pete.scott@star-dundee.com

Stephen Mudie
STAR-Dundee
Dundee, Scotland, UK
stephen.mudie@star-dundee.com

Steve Parkes
STAR-Dundee
Dundee, Scotland, UK
steve.parkes@star-dundee.com

***Abstract*—Although SpaceWire [1] is fast approaching its 20th anniversary, the requirements for equipment to test and develop new SpaceWire devices and networks continues to evolve and push the capabilities of commercial technology.**

STAR-Dundee has developed a new PCI Express product, the SpaceWire PCIe Mk2, to take advantage of the higher data rates offered in newer generations of the PCIe standard. The device features three SpaceWire ports capable of operating at 400 Mbit/s link speeds, and a PCIe Gen-3 interface that enables transmitting and receiving from/to software at the maximum rate on all links concurrently.

This high throughput is achieved through improvements to the STAR-System software suite [2], which also includes other new features to support the latest requirements for SpaceWire test and development. As well as new graphical applications providing functionality such as the ability to operate as a Remote Memory Access Protocol (RMAP) [3] initiator, support for ARM targets and the Python scripting language is also included.

This paper describes the advancements present in the SpaceWire PCIe Mk2 board and the STAR-System software suite, which simplify test and development activities while also providing higher throughput and lower latency transmission and reception of data.

Keywords—SpaceWire, Test and Development, SpaceWire PCIe Mk2, STAR-System

I. Introduction

The requirements for SpaceWire test and development products continue to evolve as the functionality offered by commercial technology changes. Back in 2002, a SpaceWire PCI board with ports operating at 200 Mbit/s, offering a maximum throughput of 800 Mbit/s to a 32-bit x86 Windows XP PC and accessed via a C API (Application Programming Interface), was a perfectly good solution for testing new SpaceWire developments.

In 2022, 400 Mbit/s SpaceWire links are sometimes required, while PCIe Gen-3 offers the capability to transfer data to and from all three of the SpaceWire ports on STAR-Dundee's new PCIe board, the SpaceWire PCIe Mk2, at the full 400 Mbit/s data rate on each port, giving a total throughput in excess of 1.8 Gbit/s. The host computers have also evolved, and STAR-Dundee's latest STAR-System software suite supports both Windows and Linux on 32-bit and 64-bit x86 machines, with ARM targets also supported on Linux.

The C programming language continues to be popular but newer scripting languages such as Python simplify the implementation of test scripts, while in some scenarios graphical applications to perform common tasks can potentially eliminate the need for any development using an API. The latest version of STAR-System, version 5.01, includes C, C++ and Python APIs plus several new graphical applications to perform tasks such as configuring deterministic triggering behaviour and acting as an RMAP initiator.

This paper describes the many benefits of the SpaceWire PCIe Mk2 combined with the STAR-System software suite, including the high performance it offers for SpaceWire test and development.

II. The SpaceWire PCIe Mk2

The SpaceWire PCIe Mk2 board was developed to replace the SpaceWire PCIe (Mk1) board, which was initially released in 2012. Like its predecessor, the PCIe Mk2 has three SpaceWire ports, which can be used to transmit and receive data from/to software at high speed, and it can act as either an interface or a router. A photograph of the SpaceWire PCIe Mk2 is shown in Figure 1.

Unlike its predecessor, the SpaceWire ports on the PCIe Mk2 can operate at link speeds of up to 400 Mbit/s and there are two external SMB trigger interfaces which can be configured as input or output triggers. There is also extensive fault protection to meet most FMEA (Failure Mode and Effects Analysis) compliance requirements. This protection covers the input power voltages from the host PC, the overvoltage of any of the point of load converters on the board, the output voltage on the SpaceWire ports, and the trigger output voltage. The SpaceWire ports are cold-sparing so that the PCIe Mk2 board can be powered down without adversely affecting any system it is connected to by a SpaceWire link, while the SpaceWire LVDS (Low-Voltage Differential Signalling) transmitters can be tri-stated.

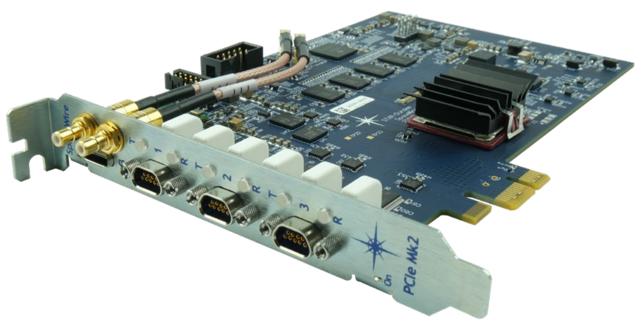


Fig. 1. SpaceWire PCIe Mk2

The PCIe interface to the host PC is a Gen-3, ×1 lane which is compatible with Gen-1, Gen-2, Gen-3 and Gen-4 PCIe slots of ×1, ×4, ×8 and ×16 widths, making it incredibly flexible. This interface also provides the data rates required to support all three SpaceWire ports transmitting and receiving data concurrently.

III. STAR-System

The software provided with the SpaceWire PCIe Mk2 hardware is the STAR-System software suite which supports all STAR-Dundee's SpaceWire and SpaceFibre interface and router devices released since 2011, including the original SpaceWire PCIe (Mk1). This helps to ensure backwards compatibility between the two products, with the same software interfaces provided to access both the Mk1 and Mk2 PCIe boards, and STAR-Dundee's other products.

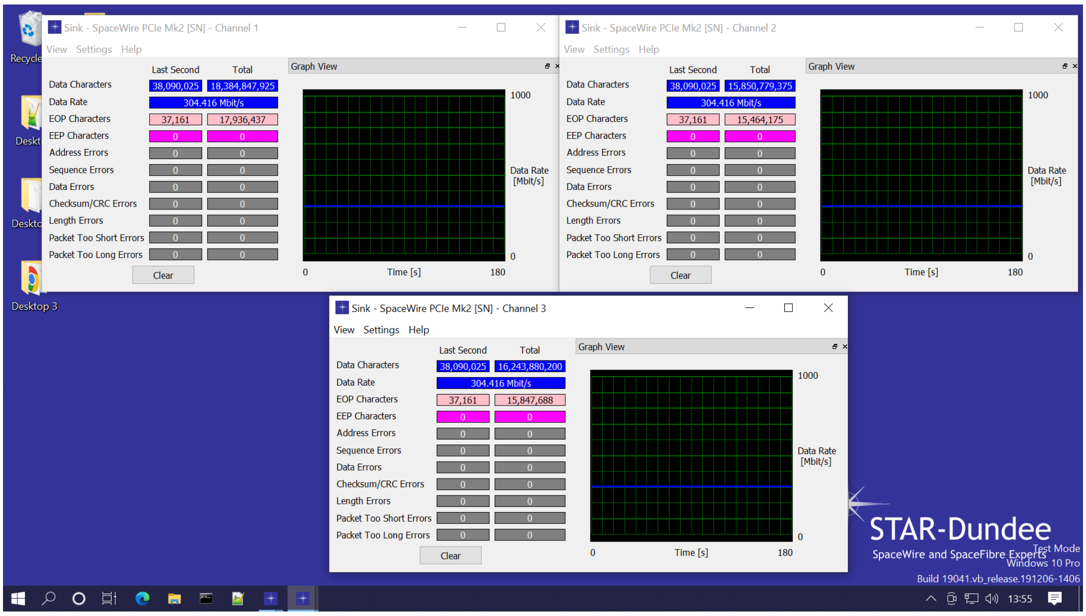
The latest release of STAR-System, version 5.01, has been updated to include support for the SpaceWire PCIe Mk2 while continuing to support all previous devices. A new driver for the device, designed to also support higher speed SpaceFibre [4] devices, offers very high performance. Several new features have also been added, with the inclusion of a new API to support CCSDS Space Packet Protocols [5] along with a version of the APIs for the Python scripting language, which can be a powerful means to script tests, for example.

Support has recently been added to the Linux release of STAR-System for ARMv6, ARMv7 and ARMv8 targets, in addition to the previously supported i386 and x86-64 targets, as more development and testing is conducted using ARM processors. The latest release has also been successfully tested on Windows 11 and support has been added for the latest Linux kernel at the time of release, v5.16.9.

There have been many improvements to the graphical applications included with STAR-System. These provide functionality commonly required during test and development, not only to transmit and receive packets but also to inject errors and access the triggering functionality, for example. The latest updates include a new application that can act as an RMAP initiator and the addition of graphs to the Source and Sink applications which can transmit and receive packets at very high rates. The screenshot in Figure 2 shows three Sink windows receiving packets in a triple loopback test at the theoretical maximum data rate for a 400 Mbit/s link of around 304 Mbit/s per port, graphing those rates for each port.

As previously mentioned, STAR-System and the PCIe Mk2 are capable of much more than simply transmitting and receiving packets. An Error Injection application can be used to inject different types of errors on the link, while corresponding API functionality can transmit these errors in sequence with data characters to ensure an error occurs at a defined point in a packet. Packet timestamps can be added to received packets so that the start and end time of each packet can be recorded at sub-microsecond resolution.

In addition to the external trigger interfaces on the front of the device, the PCIe Mk2 includes further triggering capability which can be accessed from the STAR-System Triggering API or the associated graphical application. This triggering capability can be used to trigger an action to be performed when an event occurs. The event may be a signal on one of the external trigger interfaces, or it may be a packet being received or an error occurring on the link. The action may be to transmit a packet or time-code or to signal to on one of the trigger interfaces. Several different events and actions are supported, including counters which can be used to ensure an action occurs at a specific time. These actions and events can then be combined to provide deterministic behaviour, even when using a non-deterministic operating system, with the hardware responsible for determining when an action should take place.



Sink - SpaceWire PCIe Mk2 [SN] - Channel 1

	Last Second	Total
Data Characters	38,090,025	18,384,847,925
Data Rate	304.416 Mbit/s	
EOP Characters	37,161	17,936,437
EEP Characters	0	0
Address Errors	0	0
Sequence Errors	0	0
Data Errors	0	0
Checksum/CRC Errors	0	0
Length Errors	0	0
Packet Too Short Errors	0	0
Packet Too Long Errors	0	0

Sink - SpaceWire PCIe Mk2 [SN] - Channel 2

	Last Second	Total
Data Characters	38,090,025	15,850,779,375
Data Rate	304.416 Mbit/s	
EOP Characters	37,161	15,464,175
EEP Characters	0	0
Address Errors	0	0
Sequence Errors	0	0
Data Errors	0	0
Checksum/CRC Errors	0	0
Length Errors	0	0
Packet Too Short Errors	0	0
Packet Too Long Errors	0	0

Sink - SpaceWire PCIe Mk2 [SN] - Channel 3

	Last Second	Total
Data Characters	38,090,025	16,243,880,200
Data Rate	304.416 Mbit/s	
EOP Characters	37,161	15,847,688
EEP Characters	0	0
Address Errors	0	0
Sequence Errors	0	0
Data Errors	0	0
Checksum/CRC Errors	0	0
Length Errors	0	0
Packet Too Short Errors	0	0
Packet Too Long Errors	0	0

Fig. 2. STAR-System Sink Showing Triple Loopback Results

IV. Performance

Even when not using the triggering functionality, the PCIe Mk2 and STAR-System have been designed to offer extremely high performance when combined. This not only includes providing high throughput and low latency, but also ensuring CPU usage on the host PC transmitting and receiving packets is kept low, to allow further processing of the data being transmitted and received to be performed.

STAR-System includes several tools to test the performance of a product and/or the devices that they are connected to, including the Source and Sink applications mentioned earlier. The STAR-System Performance Tester application, however, provides the most comprehensive performance testing, allowing throughput and latency tests to be performed and the results output to a text file for graphing in a spreadsheet.

To measure the performance of the PCIe Mk2, several tests were performed using the Performance Tester on a relatively low-cost test PC with the following specifications:

- ASUS PRIME H310i PLUS R2.0 Motherboard
- Intel Core i5 Six Core Processor i5-9600 (3.1GHz) 9MB Cache
- 8 GB Corsair VENGEANCE DDR4 2400MHz
- 240 GB ADATA SU630 SSD

This PC is dual-bootable, allowing tests to be performed on 64-bit versions of both Windows 10 and Linux. The results of the tests were consistent between the two platforms as STAR-System has been designed to be efficient on each.

The most basic test is a throughput test with the device in loopback, where a SpaceWire cable is connected between two ports of the PCIe Mk2. The Performance Tester allows such a test to be repeated for a range of different packet sizes and the chart in Figure 3 shows the data rate and CPU usage for packet sizes between 1 and 100 bytes along the x-axis.

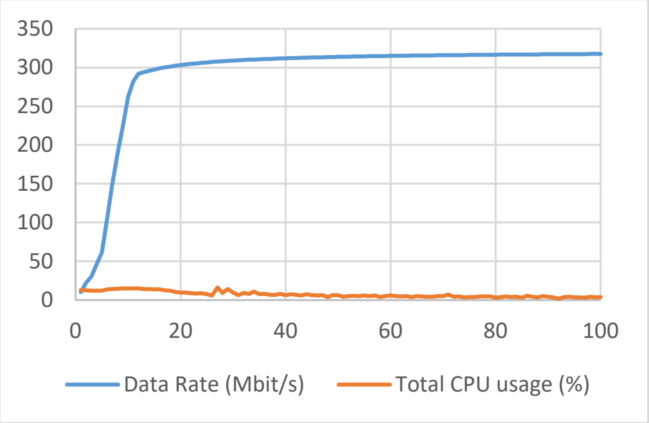


Fig. 3. Single Loopback Test on Windows

For this test, the Windows operating system was used, and the link speed was set to 400 Mbit/s to test the fastest possible data rate with the board. On a 400 Mbit/s link, the theoretical maximum for an infinite length packet is 320 Mbit/s due to SpaceWire's use of 10 bits to encode each byte of data. End of Packet markers also reduce the maximum data rate which can be achieved when sending packets, and the chart shows that STAR-System and the PCIe Mk2 achieve a data rate close to the theoretical maximum for packet sizes of 12 bytes and above. CPU usage is also below 15% for most of the test, ensuring that it's possible to do more than just transmit and receive data but also process the received data with the available CPU time.

Figure 4 shows the results of a similar test, but this time on the Linux operating system and using all three ports of the PCIe Mk2. Ports 1 and 3 are connected with a SpaceWire cable, while a loopback cable is connected to port 2. This allows all three SpaceWire ports to be exercised concurrently with packets flowing in both directions on each port. Due to the additional overhead of flow control tokens, which are sent in the opposite direction to the data, the theoretical maximum on each port operating at 400 Mbit/s is reduced from 320 Mbit/s to around 304 Mbit/s, giving a total theoretical maximum of around 912 Mbit/s.

With this test, it takes slightly longer to approach the theoretical maximum, but does so for packet sizes greater than 50 bytes in length. Note that as packets are flowing in both directions on each port, the total throughput for transmitting and receiving combined is twice what is shown in the chart, resulting in a total throughput exceeding 1.8 Gbit/s. As with the single loopback test, the triple loopback test does not make full use of the CPU with usage around 40% for the smallest packets and dropping to less than 20% for 100-byte packets. This should give plenty of capacity to process packets without affecting throughput.

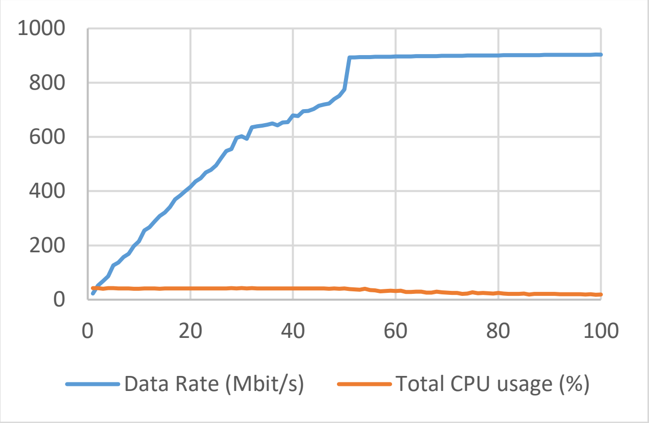


Fig. 4. Triple Loopback Test on Linux

The Performance Tester's latency test was also used to measure the average time to transmit and receive a packet. The chart in Figure 5 shows the average time in microseconds to transmit a packet from software out of one port of the SpaceWire PCIe Mk2 and receive it on another port of the PCIe Mk2 into software. This average latency is therefore the average round-trip time for a single packet. The test was performed under the Windows operating system on the same PC as the other tests, once again using packet sizes from 1 to 100 bytes and a link speed of 400 Mbit/s.

The chart shows that there is very little overhead introduced by software or the PCIe Mk2 to the round-trip time. The difference in latency between a packet of 1 byte in length and one of 100 bytes in length is around 2 or 3 microseconds, which can be attributed to the additional time it takes for the larger packet to travel over the SpaceWire link. The overhead introduced by STAR-System and the PCIe Mk2 is therefore around 48 microseconds.

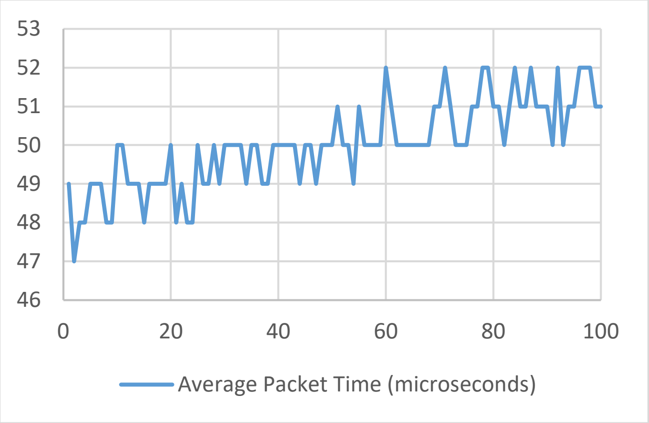


Fig. 5. Latency Test on Windows

However, it should be remembered that this is a worst-case scenario. STAR-System and the PCIe Mk2 are designed to be incredibly efficient when transmitting and receiving multiple packets, so this overhead will not apply to each individual packet. Instead, it's likely that the total overhead for groups of packets will be similar, with only the additional time required to cross the SpaceWire link added.

V. Conclusions

The SpaceWire PCIe Mk2 is the fastest SpaceWire product developed so far by STAR-Dundee. Its Gen-3 PCIe interface, support for 400 Mbit/s SpaceWire link speeds and integrated software and hardware design, permits total throughput exceeding 1.8 Gbit/s. This is achieved with relatively low CPU usage, allowing the generation of data to be transmitted and the processing of data received to be conducted in parallel.

The STAR-System software suite includes many of the APIs required for this generation and processing including individual APIs for commonly used protocols such as RMAP and the CCSDS Space Packet Protocol. Graphical applications provide the capability to perform many of these tasks without any programming, while a Python API and optional LabVIEW support [6] can be used to quickly script tests.

This is all achieved while maintaining backwards compatibility with the original SpaceWire PCIe board and with the same interface provided with other STAR-Dundee interface and router devices. This makes migrating from another product to the PCIe Mk2 a very simple exercise.

Through the use of a common interface and devices which support field upgrading, STAR-Dundee's products can also evolve to include new features or to support new targets as the commercial computing environment changes. Version 5.01 of STAR-System has support for the latest versions of Windows and Linux, with ARM support included in the Linux release. New features continue to be developed for both STAR-System and the PCIe Mk2 and are made available through STAR-Dundee's website to existing users, ensuring the PCIe Mk2, and the Mk1 version of the product, will continue to provide the capabilities required for SpaceWire test and development for many years to come.

The SpaceWire PCIe Mk2 is already in production and began shipping to users in September of 2022.

Acknowledgements

STAR-Dundee would like to acknowledge the feedback provided by the users of its products, which act as requirements for improvements to existing products and features of new products.

References

[1] ECSS, "SpaceWire – Links, nodes, routers and networks", Standard ECSS-E-ST-50-12C Rev.1, European Cooperation for Space Standardization, May 2019, available from http://www.ecss.nl.

[2] S. Mills and S. Parkes, "A Software Suite for Testing SpaceWire Devices and Networks", Proceedings of Data Systems in Aerospace (DASIA) Conference, Barcelona, Spain, 2015.

[3] ECSS, "SpaceWire – Remote memory access protocol", Standard ECSS-E-ST-50-52C, Issue 1, European Cooperation for Space Standardization, February 2010, available from http://www.ecss.nl.

[4] ECSS, "SpaceFibre – Very high-speed serial link", Standard ECSS-E-ST-50-11C, Issue 1, European Cooperation for Space Starndardization, May 2019, available from http://www.ecss.nl.

[5] CCSDS, "Space Packet Protocol. Recommendation for Space Data System Standards (Blue Book)", CCSDS 133.0-B-1, Issue 1, Consultative Committee for Space Data Systems, September 2003, available from https://public.ccsds.org/.

[6] STAR-Dundee, "STAR-System for LabVIEW", https://www.star-dundee.com/products/star-system-labview, STAR-Dundee Website, accessed September 2022.

SpaceWire Codec VIP: an innovative architecture of UVM-based Verification Environment

SpaceWire Test and Verification, Short Paper

Simone Vagaggini
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa – ITALY
simone.vagaggini@phd.unipi.it
IngeniArs S.r.l
Via Ponte a Piglieri 8, I-56121, Pisa - ITALY
simone.vagaggini@ingeniars.com

Marco Trafeli
IngeniArs S.r.l
Via Ponte a Piglieri 8, I-56121, Pisa - ITALY
marco.trafeli@ingeniars.com

Roberto Ciardi
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa – ITALY
roberto.ciardi@phd.unipi.it
IngeniArs S.r.l
Via Ponte a Piglieri 8, I-56121, Pisa - ITALY
roberto.ciardi@ingeniars.com

Daniele Davalle
IngeniArs S.r.l
Via Ponte a Piglieri 8, I-56121, Pisa - ITALY
daniele.davalle@ingeniars.com

Lucana Santos
European Space Agency, ESTEC
Keplerlaan 1, 2201 AZ, Noordwijk – NETHERLANDS
Lucana.Santos@esa.int

Pietro Nannipieri
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa - ITALY
pietro.nannipieri@unipi.it

Luca Fanucci
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa - ITALY
luca.fanucci@unipi.it

Abstract— **SpaceWire (SpW) devices are widely used in space applications on-board satellites. Their verification is fundamental because it ensures that the Design Under Test (DUT) is bug-free, without the risk of compromising an entire space mission.**

In this paper an innovative architecture of a Verification Intellectual Property (VIP) based on Universal Verification Methodology (UVM) for the testing of a SpW Codec IP core is presented. Its core is the Twin Model, developed in SystemVerilog and compliant with UVM, which emulates the ideal behavior of the SpW Codec. It communicates directly with the DUT through the Data-Strobe interface with the advantage of automatically generating and sending the data needed to create and maintain the link and leaving only the definition of the payload data to be transmitted to the user. Other two Twin Models are used to create a secondary communication link, named Twin Link, in parallel with the main one. This is the best solution to verify the correct behavior of the DUT in case of errors on the link. In fact, since the Twin Link emulates the ideal communication link behavior, it provides a benchmark for verifying the one including the DUT. A specific functional block is designed to inject on the link all types of errors defined by the standard. The result is a highly reliable and configurable VIP that allows for automatic testing of all the functionalities of any SpW codec. Finally, a full verification campaign was performed with two different DUTs, achieving 100% functional coverage.

Keywords— Universal Verification Methodology (UVM), Space-Wire (SpW) Codec, Verification IP, Twin Model, Twin Link, Verification Environment, functional verification, reusability, SystemVerilog.

I. INTRODUCTION

SpaceWire (SpW) [1] is a widely used communication protocol for both direct and indirect interconnection of two or more devices, especially for aboard satellites in space missions. Because of its area of use, the verification phase assumes a crucial role: once the system has been launched into space, any error or malfunction would not be fixable and could compromise the entire mission. Therefore, it is essential to ensure that each device is extremely reliable and bug-free.

The huge number of possible scenarios to be tested makes verification very critical, especially with traditional methods such as creating ad hoc stimuli and manually observing if the actual behavior matches the expected one. For example, before information data transmission can begin, two SpW hosts must complete a standard-defined handshake protocol, which is used to initialize the communication link. Once initialized, specific codes such as Flow Control Tokens (FCTs) and NULL codes are exchanged in addition to the payload for controlling the flow and for keeping the communication active. This would require continuous adaptation of the data to be transmitted and the expected one according to the instant-by-instant state of the host which communicates with the Design Under Test (DUT).

However, advanced verification methodologies allow this problem to be avoided: one example is the development of a Twin Model that emulates the ideal behavior of a SpW node and is able to communicate directly with the DUT through the serial Data-Strobe interface [2]. By integrating it within an advanced and automated verification environment, it allows only the information data to be defined and sent, since the codes for controlling and maintaining the communication are generated and sent automatically.

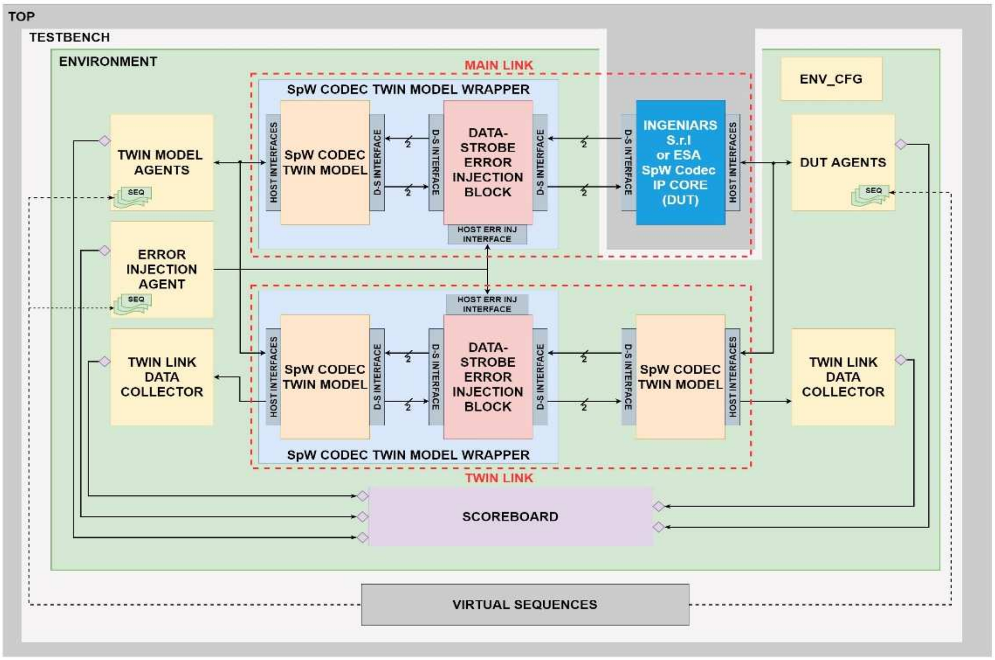


Fig. 1 Verification Environment High-Level Block Diagram

Unfortunately, a direct communication between DUT and Twin Model is not sufficient to verify effectively all the requirements defined by the standard. In fact, during the communication each SpW node performs continuous checks on the received codes and on the status of the other host, and if an error is detected the link is interrupted. So, with the single link architecture there is no possibility to check whether the DUT has the correct behavior in a pseudo-random error scenario, and a manual inspection would be required. This approach would not be efficient neither reliable and would require a lot of time to check all possible error scenarios.

In this paper a Verification Intellectual Property (VIP) that can be used for testing any vendor's SpW codec is presented. It implements a verification environment developed in the SystemVerilog Hardware Verification Language (HVL) [3] and is fully compliant with the Universal Verification Methodology (UVM) standard [4][5][6] that represents the current state-of-the-art for verification.

It puts forward a totally innovative architecture: a direct communication between DUT and REF makes up the main communication link and in parallel a second communication link called "Twin Link" and consisting of two Twin Models is implemented. This emulates the ideal behavior of the SpW link in case of errors, providing a benchmark against which the actual behavior of the DUT can be compared.

The verification environment includes also a functional block for the injection of all types of errors foreseen by the SpW standard on the Data-Strobe interface and it can be easily controlled by the user.

The paper is organized as follows:

- Section II presents the Verification Environment, with particular focus on the Data-Strobe error injection block, on the Twin Model and on the Twin Link.
- Section III describes the Verification Environment has been used for a comprehensive verification campaign of a SpW Codec, with particular focus on how the testcases are structured and easily configured exploiting the power of SystemVerilog and UVM.
- Section IV summarizes the results and conclusions.

II. UVM Verification Environment

In this section an overview of the Verification Environment, with a focus on the Data-Strobe error injection block, on the Twin Model and on the Twin Link is given. The high-level block diagram of the verification environment is shown in Fig. 1.

A. Error Injection Block

This functional block was developed in SystemVerilog and conforms to UVM. As shown in the high-level block diagram of Fig. 2, it should be instantiated between two SpW nodes, connecting it to the Data-Strobe interfaces in both directions. Through a dedicated interface, it is possible to apply for error injection in a specific direction. If no error is requested, it acts as a simple link between input and output. The errors that can be injected are:

- Parity Error: during transmission of the current Normal Character (Nchar) or code, the parity bit is corrupted.
- Disconnect Error: both data and strobe are frozen for at least 850ns.

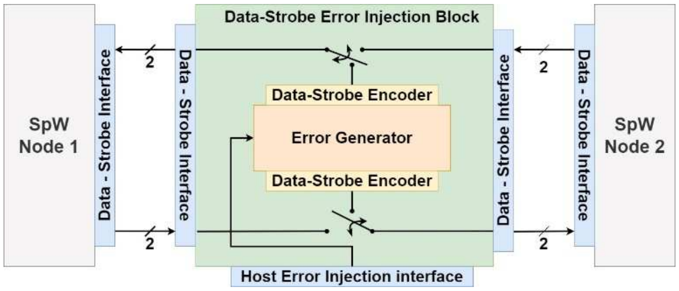


Fig. 2 Data-Strobe Error Block

- ESC error: Instead of sending the current Nchar or code, an Escape (ESC) code followed by an ESC or a End of Packet (EOP) code or an Error End of Packet (EEP) code is transmitted.
- Credit Error: Instead of sending the current Nchar or code, FCTs are transmitted until it causes an overflow in transmission credit counter of the other SpW node.
- GotFCT/GotNChar/GotFCT Errors: Instead of sending the current Nchar or code the transmission of an FCT/Nchar/Broadcast Code (BC) are forced.

Note that it is possible to inject errors at any instant in order to create all possible error scenarios defined by the SpW standard.

B. SpaceWire Codec Twin Model

The Twin Model is the core of the verification environment because it emulates the ideal behavior of a SpW codec and communicates with the DUT [2]. It is developed in SystemVerilog HVL and it is fully compliant with UVM standard. To fully benefit from the power of UVM-based approach in terms of reusability and maintainability, it has been designed internally with three communicating layers, reflecting the ones described by the standard: Network Layer, Data-Link Layer, and Encoding Layer.

The use of the Twin Model significantly simplifies the definition of the simulation scenario, because it automatically performs the operations required for link initialization, it transmits NULL codes when required to avoid disconnections and it handles the transmission and reception credit counters.

As shown in the block diagram in Fig. 3, taking advantage of UVM's feature of extending a functional block, a Twin Model Wrapper is created. A complete model is obtained combining the Twin Model functionality with the capability to inject errors.

C. Agents

They are used to stimulate and monitor both the DUT and the Twin Model Wrapper. After the stimulation implemented by the driver functional block, the agent is responsible for notifying the scoreboard of all events observed on the interfaces.

There are 20 agents to perform the following operations at the DUT and Twin Model:

- To provide clock and reset signals.
- To request the transmission and enable the reception of Nchars packets, Time-Codes, Interrupts and Interrupt Acknowledgements.
- To change or read the device configurations.

There is also an agent used to request the error injection to the dedicated functional block.

D. Double Link

As shown in Fig. 1, a second link, called the Twin Link, is designed in parallel with the main one. It connects a Twin Model Wrapper with a Twin Model, where the second one acts as reference model for the DUT. The stimuli required for the main link are also sent to the Twin link. Since the Twin Link is composed of two ideal models, it provides a reference against which the behavior of the main one, conditioned by the DUT, can be compared.

Without the reference link, it would be impossible to understand in an automated way whether the DUT has the expected behavior after a pseudo-random error. In fact, in that case an inspection of the DUT's internal signals would be required, thus being inefficient and extremely time-consuming. Instead, this innovative architecture allows this problem to be solved and limits the control to be performed to a

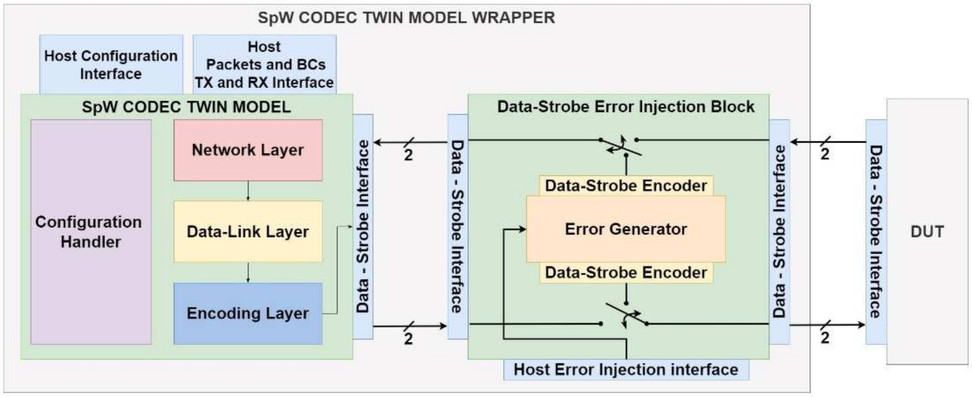


Fig. 3 SpW Codec Twin Model connected to Data-Strobe Error injection Block

synchronization and comparison of the flow on the main link with that on the Twin Link. Since the SpW standard [1] defines more than 30 different scenarios involving errors it is evident that this architecture has significant advantages in terms of reliability and cost.

Note that the Twin Link is not necessary in error-free simulation scenarios, in which a single line of communication between the DUT and Twin Model is sufficient to verify the correct operativity: in fact, it is only necessary to verify that the packets and BCs sent and received by the DUT match those received and sent by the Twin Model, respectively, and that both changes and readings of configuration are handled successfully. For this reason, the Twin Link instance can be enabled or disabled by the testcase depending on the desired scenario lightening the simulator overhead and reducing simulation time.

III. Verification Campaign and Results

A comprehensive verification campaign has been done using this VIP: all the testcases necessary to achieve 100% functional coverage have been defined.

A. Device Under Test (DUT)

Two different and unrelated DUTs have been tested, as evidence that this Verification Environment is suitable for any SpW Codec. Note that for some SpW Codecs an adapter could be necessary for the interfaces with the host because they are not univocally defined by the standard [1], resulting to be implementation-dependent.

The first one is the SpW Codec IP Core developed by IngeniArs S.r.l [7] that is compliant with the revision 1 of the ECSS-E-ST-50-12C standard. The second one is the SpW Codec IP Core of European Space Agency (ESA) [8], compliant with the first release of ECSS-E-ST-50-12C standard.

B. Testcases

The hierarchical structure of UVM allows the testcase to configure the verification environment according to the desired scenario, for example, by enabling or disabling the instantiation of certain blocks.

The customization of stimuli is done through the mechanism of virtual sequences which are used to create sequences of transactions. Their configuration is very easy by only the setting of parameters.

All testcases necessary to hit all the standard-defined requirements (100% functional coverage) have been implemented. Due to the great flexibility of this system, new testcases can be designed with a very small effort to achieve other goals, such as in terms of code coverage.

IV. Conclusions

A Verification IP capable of executing a full test campaign on any SpW Codec was presented.

It is based on a Twin Model of the SpW Codec that when connected to any SpW device through the Data-Strobe interface is able to automatically initialize the link, keep it active, and control the flow by managing transmission and reception credits. This significantly simplifies the task of the verification team because without this design it would be necessary to adjust the stimuli to be sent to the DUT depending on the instant-by-instant state of the communication, which depends on the behavior of the DUT itself.

In addition, the proposed verification environment offers a completely innovative architecture that allows for easy and efficient verification of all scenarios with errors described by the standard: it is based on a Twin Link in parallel with the main one. Being composed of two Twin Models, it provides an ideal model of the behavior of a SpW link in case an error occurs. As a result, it is possible to verify the correct functioning after an error by comparing the data flow on the main link (affected by the DUT) with the expected one.

By using a specific block capable of injecting any type of error on the Data-Strobe interface, the tests necessary to achieve 100% functional coverage were defined. The test campaign was performed on two different IP Cores, demonstrating the high flexibility and reusability of this VIP.

Last but not least, the VIP is fully compliant with UVM: thanks to the layered structure suggested by UVM, it results to be extremely user-friendly and does not require accurate knowledge of the internal architecture to be used. This leads to a separation between those who develop the VIP and those who use it for verification, similar to what happens with traditional IP cores. In addition, because of the compliance to UVM, it is possible to integrate this VIP into more complex ones for verification of systems having SpW interfaces.

References

[1] ECSS, “SpaceWire – Links, nodes, routers and networks” ESA Requirements and Standards Division, Noordwyk (NL), May 2019, P/N ECSS E 50 12C rev. 1

[2] S.Vagaggini, R.Ciardi, M.Trafeli, L.Fanucci, “Development of highly reliable UVM-based Verification Environment for SpaceWire Codec”, 2022 IEEE 9th International Workshop on Metrology for AeroSpace (MetroAeroSpace). June 2022.

[3] C. Spear, G. Tumbush, “SystemVerilog for Verification, a Guide to Learning the Testbench Language Features, third edition”, Springer Science & Business Media, 2012

[4] Standard Universal Verification Methodology.http://accellera.org/downloads/standards/uvm

[5] Mentor Graphics, “Universal Verification Methodology, UVM Cookbook”, Online Methodology Documentation from the Verification Methodology Team, 2019

[6] Doulos, “UVM Golden Reference Guide”, December 2013

[7] IngeniArs SpaceWire Codec IP Core - https://www.ingeniars.com/in_product/spacewire-codec-ip-core (Accessed on 22/08/2022).

[8] European Space Agency SpaceWire Codec IP Core - https://www.esa.int/Enabling_Support/Space_Engineering_Technology/Microelectronics/SpWb (Accessed on 22/08/2022).

Eye Diagram Analyser for Space High-Speed Serial Links: a Tool for Evaluating Signal Integrity in SpaceFibre Links.

Luca Dello Sterpaio
IngeniArs S.r.l.
Pisa, Italy
luca.dellosterpaio@ingeniars.com

Gianmarco Dinelli
IngeniArs S.r.l.
Pisa, Italy
gianmarco.dinelli @ingeniars.com

Daniele Davalle
IngeniArs S.r.l.
Pisa, Italy
daniele.davalle@ingeniars.com

Pietro Nannipieri
University of Pisa
Pisa, Italy
pietro.nannipieri@unipi.it

Luca Fanucci
University of Pisa
Pisa, Italy
luca.fanucci@unipi.it

Abstract—**An eye diagram, or eye pattern, is a valuable metric to assess the quality of a digital signal in telecommunications. It represents a remarkable solution in those cases where it is not possible to analyze the quality of the end-to-end communication by directly measuring electric signals. Nevertheless, this kind of analysis is nowadays required also in the space domain since modern on-board data-handling systems can feature multi-gigabit communication links. For example, the European Space Agency (ESA) supports the adoption of the SpaceFibre protocol, which provides a maximum data rate of 6.25 Gbps. In this paper, we present the integration of the eye margin analysis in a commercial test equipment for high-speed protocols: the SpaceWire/SpaceFibre Analyser Real-Time (SpaceART). The SpaceART instrument features Multi-Gigabit serial Transceivers (MGT), and it is compatible with the data rate of current and next generation space high-speed communication protocols. The proposed system for Electrical Ground Support Equipment (EGSE) allows users to automatically capture and visualize on a dedicated Graphical User Interface (GUI) the resulting output of the eye scan system within the seamless testing environment. Statistical Eye diagrams acquisition results are presented for several SpaceFibre data-rate of reference, highlighting the effects of different communication speeds on the SpaceFibre link signal integrity.**

Keywords—Eye Scan, SpaceFibre, WizardLink, Signal Integrity, SpaceART, EGSE.

I. Introduction

Today's space missions' requirements for on-board communication are high as never before, demanding to be capable to move large amount of data, quickly and in a reliable way. SpaceFibre (SpFi) has been standardized by the European Cooperation for Space Standardization (ECSS) in 2019 [1], and it has been designed to fulfill the data rate requirements of multi-gigabit payloads, such as Synthetic Aperture Radar (SAR) and hyper-spectral imagers. SpaceFibre links can sustain a maximum data rate of 6.25 Gbps per lane (up to 16 lanes), and it is backward compatible with the well-established SpaceWire (SpW) protocol at packet level, enhancing its capabilities in terms of data-rate, quality of service and reliability. Indeed, SpaceWire standard supports a maximum link-rate of 400 Mbps, resulting inadequate for high bandwidth instruments already necessary in several missions, such as the Multi Spectral Imager (MSI) on board of Sentinel-2 [2], TerraSAR [3] and NISAR SAR [4]. Similar issues were recently solved using several SpaceWire links in parallel or adopting other solutions such as WizardLink a proprietary protocol that does not define data-link layer and link initialization. Solutions based on WizardLink have custom implementations of the missing layers, losing the benefits of standardization. Thus, SpaceFibre aims at filling this gap, providing standardized high-speed connectivity, specifically intended for space applications. For these reasons and also considering the growing trend in the Earth Observation market [5], whose applications largely exploit high-resolution instruments [6], the analysis of high-speed links' digital signal quality represents a valuable feature for modern test equipment.

A widely adopted methodology to fulfill this task is the "*eye pattern*" diagram [7]. An eye pattern diagram provides very useful information proving both the timing and amplitude characteristics of the signal with simple visual inspection. Typically, measurement is carried out by means of a digital oscilloscope, probing the electrical contacts at receiver-end input. The resulting overlayed signal patterns of consecutive acquisitions resemble the shape of an eye: the wider the eye is open, the higher the noise margins of the signal are. At receiver pins though, the signal has not yet been enhanced by receiver-end signal-conditioning (for example, equalization); this may lead to incorrect or misleading results, potentially returning a closed eye pattern when instead it is not. Tapping the post-processed signal is usually not a viable option, if not planned by design given transceivers are manufactured directly into Integrated Circuits (IC).

To take into account RX-end signal-conditioning too, a Bit/Error Ratio (BER) statistic can be performed instead at the receiver-end. Usually, this type of measurement is often referred to as "*Statistical Eye Pattern*" diagram or, generalizing, "*RX Margin Analysis*" and its methodology, specifically related to SpFi applications, will be presented in this document. At the time this paper was redacted, no other EGSE solution was available on the market offering this type of analysis for signal integrity evaluation over SpaceFibre links, including traffic generation and analysis features, besides SpaceART units produced by IngeniArs.

This publication is organized as follows:

- Section I provides a brief, but introductory overview of the topic discussed,

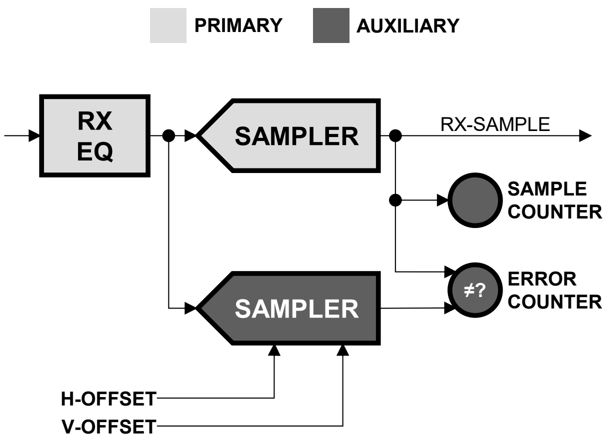


Fig. 1. RX-end diagram for Statistical Eye measurement.

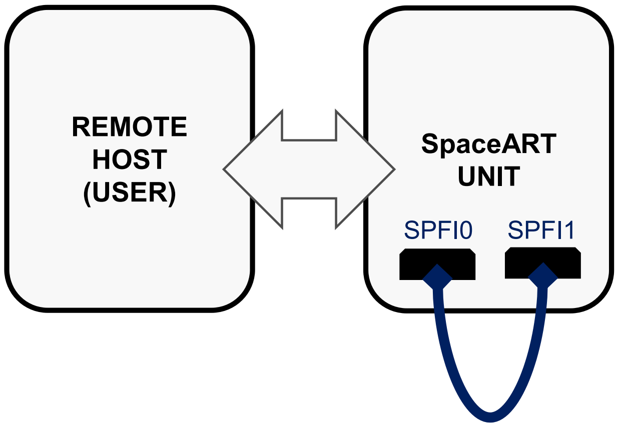


Fig. 2. Picture of testing setup.

- Section II presents the Statistical Eye Pattern theory and acquisition procedure in detail,
- Section III describes the SpaceFibre test setup for the intended RX Margin Analysis,
- Section IV presents measurement acquisitions result for in-hardware test conducted on State-of-the-Art SpaceFibre test equipment,
- Finally, Section V draws appropriate conclusions from premises and results that have been presented.

II. Statistical Eye Pattern Diagram Theory

In RX Margin Analysis, a "*Statistical Eye Pattern*" (StatEye) diagram is a BER heatmap evaluated for several – ideally all – possible receiver-end sampling points. In this case, BER is intended as the ratio between the amount of RX-data in a given sampling point that does not match the RX-data sampled from the same signal at the nominal sampling point and the total amount of received data. The shape of a StatEye resembles the one of a typical eye pattern acquired with an oscilloscope, and it contains indeed, aggregately, the very same information about signal timing and amplitude characteristics, although it is, in the end, an expression of the effects of signal quality in terms of above-mentioned BER. Namely, the perspective is overturned: we look at the effects on the intelligibility of the signal instead of on the signal itself and evaluate its intelligibility upon its shape characteristics.

Incoming differential waveform carrying serial data is recovered after RX equalization by primary sampler. To evaluate BER in a single point of the heatmap, it is necessary to sample the incoming signal twice and, thus, employ an additional sampler at RX-end. The sampling point of this additional secondary sampler will be adjusted incrementally to sweep all possible offsets (both timing and amplitude offsets) from the nominal operating point at which the main primary sampler will continue to operate (as shown in Fig. 1). Note that, by employing an auxiliary sampler, all SpFi communications over the link will then be able to carry on without being affected by the measure in any way; quite the contrary, for BER measurement purposes it is necessary to ensure through the communication across the link that the signal will swing and perform all possible transitions (which is guaranteed by the 8B/10B transmission-code anyway [8][9]). BER calculation for each sampling point is accomplished by comparing the received bits from the two samplers (within the assumption the primary sampler's bits as correct) and accumulating in two counters the number of the total received samples and how many of them are incorrect (i.e., do not match). Of course, it is necessary to collect a large number of samples for the resulting BER value to be statistically significant.

Afterward, as the sampling point is varied in terms of timing or amplitude offset, the procedure for BER calculation described previously is repeated, and again, until all possible offsets have been explored.

Equation (1) expresses how BER is intended in this context and how its value can be calculated for each sampling point once the acquisition is completed:

$$BER_{StatEye} = \log_{10}\left[\frac{N_{errors}}{N_{samples}}\right] \tag{1}$$

The range of timing offset (horizontal offset) to be explored spans from one-half bit-time early to one-half bit-time late, that is, from -0.5 to 0.5 Unit Interval (UI). The actual range of amplitude offset (vertical offset) instead may vary depending on the device and it usually spans a range of a few hundred millivolts above and below the central "*neutral*" value.

III. SpaceFibre RX Margin Analysis Setup

SpaceWire/SpaceFibre Analyser Real-Time (SpaceART) system is a complete test equipment for space-oriented high-speed links. SpaceART [10][11][12] supports both the SpaceFibre and SpaceWire standards. In particular, it provides up to four SpaceWire interfaces and two SpaceFibre interfaces. Each SpaceFibre port feature a fully-compliant SpFi CODEC end-node [13][14][15], supporting up to eight Virtual Channels (VC). A single SpaceART unit can operate as EGSE system, being able to generate, consume and process SpW/SpFi data packets in real-time. SpaceART can also act as link analyser, monitoring the traffic over a high-speed link. It includes a trace memory to analyse protocol-specific features such as frame retransmission and flow control, also allowing to trigger specific words on interface input/outputs ports. Finally, SpaceART provides an error-injection and word-replacement capability for fast and easy conformance testing of the devices under test. The SpaceART Graphical User Interface (GUI) has been updated and extended to manage the complete set of functionalities described above, with to simplify the system usage for users. Given all the above-mentioned features and characteristics, the SpaceART unit appears as the optimal platform for benefiting also of the newly developed Statistical Eye Diagram functionality. For end-user convenience, the functionality is implemented in the SpaceART system and testing environment seamlessly, in order to fully automate the long sequence of operations described in Section II for StatEye evaluation. Overall utilization required for the added feature in terms of resources of SpaceART on-board FPGA [10][11][12] is summarized in TABLE I. :

TABLE I. EYE SCAN IMPLEMENTATION FOOTPRINT

Table Head	FPGA Resources		
	Utilization	*Available*	*Utilization %*
LUT	1502	218600	0.69
Flip-Flop	1145	437200	0.26
Block RAM	1	545	0.18

The testing setup is shown in Fig. 2. Test aims to evaluate the SpFi link quality in a loopback configuration of a SpaceART unit: SpaceFibre interfaces of the device shall thus be connected together and then an active link established between them. It is not strictly necessary since SpFi active links exchange control data anyway, yet random data shall be generated and consumed at nodes in both directions during this test. Eye scans will be performed over a range of 1 IU (from -0.5 to 0.5 IU) horizontally and a range of 240 mV (from -120 to 120 mV) vertically with a resolution of 32 voltage levels of a 7.5 mV step each.

The test shall be repeated for several different data-rate values in the set {2.5, 3.125, 5.0} Gbps. This way it will be possible to evaluate how the eye pattern with increasing data-rate.

The anticipated results of the tests will show that, as the x-axis is normalized to the bit-time UI, the eye pattern amplitude opening goes decreasing as the data-rate increases. Regardless, as pass-criteria, each eye pattern resulting from the acquisitions should be compared with the reference mask defined by SpaceFibre standard [1]. It is this direct comparison that will ultimately determine the success or failure of the individual test.

IV. TEST RESULTS

Acquisition results are shown in Fig. 3, Fig. 4 and Fig. 5, ordered by increasing data-rate. Each BER heatmap shown has been overlayed with the reference SpFi eye pattern mask (in figures, shaped as a hexagon or a diamond according to reference data-rate [1]).

All resulting Statistical Eye diagrams appear to be compliant with the SpaceFibre standard mask boundaries. Nevertheless, the test brought the added value of providing a qualitative judgment, highlighting the margin of signal integrity.

The eye acquired for 5 Gbps data-rate is more shuttered, as expected, yet the SpaceFibre reference mask is also narrowed, contracted from a hexagon to a diamond according to the standard [1].

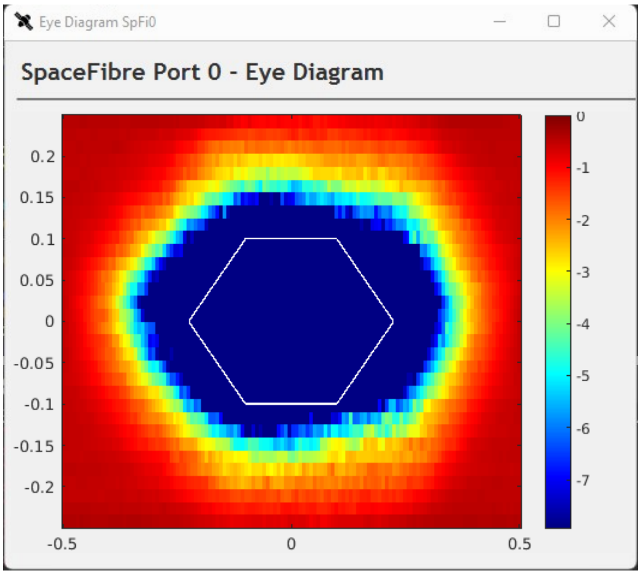


Fig. 3. Statistical Eye pattern diagram, 2.5 Gbps.

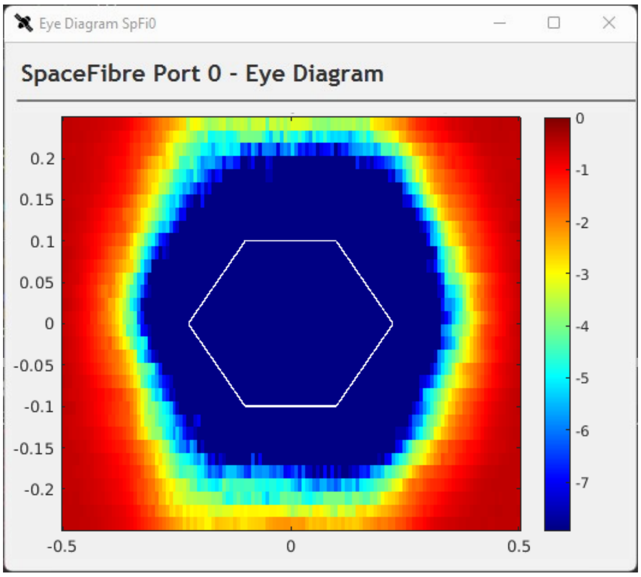


Fig. 4. Statistical Eye pattern diagram, 3.125 Gbps.

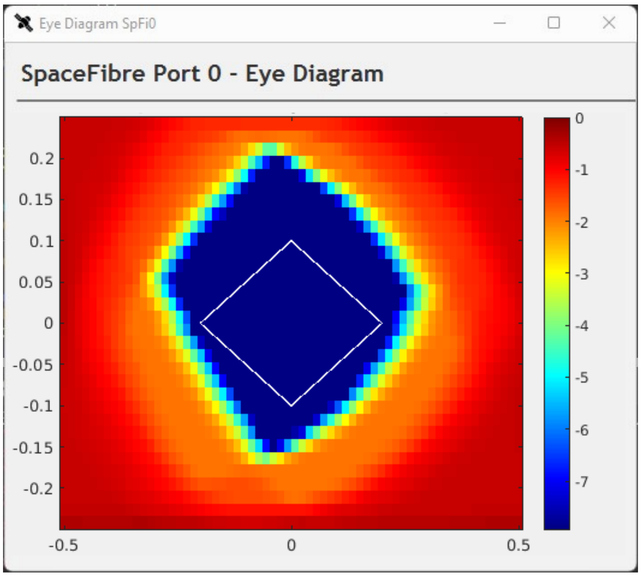


Fig. 5. Statistical Eye pattern diagram, 5 Gbps.

V. Conclusions

This paper presents the innovative application of Statistical Eye Diagram analysis as a tool for signal integrity specifically for SpFi links.

After a brief introduction, Section I provided reasons and real-world examples why inspecting signal integrity for protocol links with data-rates in the range of units of Gbps is a factual need to be addressed in today's space missions; the theory behind the Statistical Eye Diagram measurements for SpFi links was presented in Section II; Section III outlined the configuration setup, the intended purpose and the expected results of the planned tests, whose actual results acquired for several data-rates were presented and discussed in Section IV.

In conclusion, thanks to the newly developed Eye Scan functionality, SpaceART by IngeniArs now has been further enhanced as an EGSE tool. This resourceful feature allows end users to evaluate the signal integrity of up to two SpaceFibre links, simultaneously and independently. The added functionality of RX Margin analysis makes SpaceART an even more comprehensive and flexible unified EGSE solution for developing and testing next-generation devices that intend to adopt the novel SpaceFibre standard protocol.

References

[1] European Cooperation for Space Standardization, "*SpaceFibre – Very high-speed serial link,*" ECSS-E-ST-50-11C, May 2019, European Cooperation for Space Standardization.

[2] P. Martimort, V. Fernandez, V. Kirschner, C. Isola, and A. Meygret, "*Sentinel-2 multispectral imager (msi) and calibration/validation,*" International Geoscience and Remote Sensing Symposium (IGARSS), pages 6999–7002, 2012.

[3] W. Pitz and D. Miller, "*The TerraSAR-X Satellite,*" in IEEE Transactions on Geoscience and Remote Sensing, vol. 48, no. 2, pp. 615-622, Feb. 2010, doi: 10.1109/TGRS.2009.2037432.

[4] P. Xaypraseuth, R. Satish and A. Chatterjee, "*NISAR spacecraft concept overview: Design challenges for a proposed flagship dual-frequency SAR mission,*" 2015 IEEE Aerospace Conference, 2015, pp. 1-11, doi: 10.1109/AERO.2015.7118935.

[5] G. Denis, A. Claverie, X. Pasco, J.P. Darnis, B. De Maupeou, M. Lafaye, and E. Morel, "*Towards disruptions in earth observation? New earth observation systems and markets evolution: Possible scenarios and impacts,*" Acta Astronautica, 137:415–433, 2017.

[6] P.P. Mathieu and C. Aubrecht, "*Earth observation open science and innovation,*" Springer Nature, 2018.

[7] C.M. Miller, "*High-Speed Digital Transmitter Characterization Using Eye Diagram Analysis,*" 1266 Hewlett-Packard Journal 45(1994) Aug., No,4, pp. 29-37.

[8] P. Nannipieri, D. Davalle and L. Fanucci, "*A Novel Parallel 8B/10B Encoder: Architecture and Comparison with Classical Solution,*" IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences (2018), E101.A, 1120-1122, doi: 10.1587/transfun.E101.A.1120.

[9] A.X. Widmer and P.A. Franaszek, "*A DC-Balanced, Partitioned-Block, 8B/10B Transmission Code,*" IBM Journal of Research and Development (1983). 27. 440 - 451. doi: 10.1147/rd.275.0440.

[10] L. Dello Sterpaio, P. Nannipieri, A. Marino and L. Fanucci, "*Design of a SpaceWire/SpaceFibre EGSE system based on PXI industry standard,*" Proceedings of 2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace), 2019, pp. 22-26, doi: 10.1109/MetroAeroSpace.2019.8869665.

[11] L. Dello Sterpaio, A. Marino, P. Nannipieri, G. Dinelli, D. Davalle, L. Fanucci, "*A Complete EGSE Solution for the SpaceWire and SpaceFibre Protocol Based on the PXI Industry Standard,*" Sensors 2019, 19, 5013. doi: 10.3390/s19225013

[12] A. Marino, A. Leoni, L. Dello Sterpaio, P. Nannipieri, G. Dinelli, G. Benelli, D. Davalle and L. Fanucci, "*SpaceART SpaceWire and SpaceFibre Analyser Real-Time,*" Proceedings of 2020 IEEE 7th International Workshop on Metrology for AeroSpace (MetroAeroSpace), 2020, pp. 244-248, doi: 10.1109/MetroAeroSpace48742.2020.9160319.

[13] P. Nannipieri, G. Dinelli, A. Marino, L. Dello Sterpaio, A. Leoni, L. Fanucci and D. Davalle, "A serial high-speed satellite communication CODEC: Design and implementation of a SpaceFibre interface," Acta Astronauta, Volume 169, 2020, Pages 206-215, ISSN 0094-5765, doi: 10.1016/j.actaastro.2020.01.010.

[14] G. Dinelli, P. Nannipieri, A. Marino, L. Fanucci, L. Dello Sterpaio, "*The very high-speed SpaceFibre multi-lane CoDec: Implementation and experimental performance evaluation,*" Acta Astronautica, Volume 179, 2021, Pages 462-470, ISSN 0094-5765, doi: 10.1016/j.actaastro.2020.11.028.

[15] G. Dinelli, P. Nannipieri, D. Davalle and L. Fanucci, "*Design of a Reduced SpaceFibre Interface: An Enabling Technology for Low-Cost Spacecraft High-Speed Data-Handling,*" Aerospace 2019, 6, 101. doi: 10.3390/aerospace6090101

SystemVerilog UVM-based Verification Environment for a SpaceFibre Router

Lorenzo Gigli
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa, Italy
l.gigli1@studenti.unipi.it

Pietro Nannipieri
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa, Italy
pietro.nannipieri@unipi.it

Luca Zulberti
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa, Italy
luca.zulberti@phd.unipi.it

Simone Vagaggini
Dept. of Information Engineering
University of Pisa
Via Caruso 16, I-56122, Pisa, Italy
simone.vagaggini@phd.unipi.it
IngeniArs S.r.l.
Via Ponte a Piglieri 8, I-56121, Pisa, Italy
simone.vagaggini@ingeniars.com

Luca Fanucci
Dept. of Information Engineering
University of Pisa
Via Caruso, I-56122, Pisa, Italy
luca.fanucci@unipi.it

***Abstract*—The number of space missions has seen continuous growth in the last years. Accordingly, satellite communications traffic and onboard spacecraft technologies have also increased. To manage high data flows in high-bandwidth communication protocols the European Cooperation for Space Standardization has released the SpaceFibre protocol in 2019, whose features for data handling and the introduction of routing capabilities increase the complexity of the infrastructures in such networks. Consequently, a crucial implementation step towards the realisation of a satellite high speed data-handling network is the design and the verification of a routing switch at different levels. As far as the network layer is concerned, the literature and the market lack verification environments for this type of devices. This work proposes a reusable and customisable network-level verification environment for SpaceFibre routing switches, that leverages the capabilities of SystemVerilog and the Universal Verification Methodology standard. In particular, the environment can be connected with any network topology, verifying any router individually. Furthermore, our environment requires network-level compatibility with the hardware interfaces, being independent from the implemented data-layer.**

***Keywords*—SpaceFibre Router, Network-Layer Verification, Universal Verification Methodology (UVM), SystemVerilog.**

I. Introduction

The number of space missions has seen continuous growth in the last years and is expected to continue to expand as a result of the advances in digital technologies [1]. A typical application pushing toward this growth is earth observation [2], which searches for payloads capable of delivering high-resolution images with high data rate. This trend in the commercial and defence markets has caused a phenomenal increment in satellite communications traffic and onboard spacecraft instruments. In addition, because of a harsh environment such as outer space, redundancy is needed to improve reliability. This increments the number of cables and components, which means a more complex and expensive system. Hence, it was necessary to develop communication protocols capable of supporting more and more data and higher data rates combined with higher reliability and efficiency. To manage such a complex networking, the European Cooperation for Space Standardization (ECSS) released the SpaceFibre (SpFi) [3] protocol with support from all major international space agencies, companies, and research institutes. This standard increases the capabilities of its predecessor SpaceWire (SpW) [4], among them a higher data rate, a better Quality-of-Service (QoS), a better Fault Detection Isolation and Recovery (FDIR), and the possibility to use optical fibers as communication links.

A. SpaceFibre Standard Overview

Networks adopting the SpFi standard [5] use a very high-speed serial link and network technology, designed specifically for use on board spacecraft. Respect to the SpW protocol it complements its capabilities by:

- improving data rate up to $6.25\,Gbps$ per communication lane (and up to 16 lanes operating in parallel);
- implementing an innovative QoS mechanism, capable of providing concurrent bandwidth reservations, priority, and scheduled QoS, by using hardware-separated Virtual Channels (VCs);
- integrating an FDIR technique, improving system reliability and lowering the integration time;
- supporting both copper cables and optical fibre communication links, making it possible to have a further mass reduction of the spacecraft.

Moreover, SpFi is backwards compatible with SpW at the packet level allowing easy bridging: existing SpW devices can be incorporated into a SpFi network taking advantage of its performance, QoS, and FDIR capabilities.

B. Lack of Verification Environments

Given the high complexity of the SpFi standard, exhaustive verification is needed to check whether the required specifications are achieved. Due to its recent release, only the *Data-Link* layer and the below ones have been well tested as demonstrated in [2]. However, a critical section of the SpFi protocol is the *Network* layer, and thus it is mandatory to verify all its compliance.

The only products present on the market have a software (SW) approach and can verify only a point-to-point link behaviour. An example is the SpaceWire/SpaceFibre Analyser Real-Time (SpaceART) [6], that is a test equipment for the most widespread On-Board Data Handling protocols. Another solution is a simulator, named Simulator for SpaceFibre and SpaceWire Satellite OBDH Network (SHINe) [7], based on the open-source OMNeT++ [8]. It is completely written in C++ and can simulate both SpFi and SpW protocols, extracting useful metrics such as latency and bandwidth usage. SHINe also implements an advanced Hardware-In-the-Loop (HIL) mechanism to connect physical devices to the simulator. However, the communication will not happen in real-time because the simulation speed is much slower than the data rate of a real device. Apart from this and other SW approaches, there are no structured verification environments to verify the hardware (HW) of a SpFi Routing Switch, nor in literature or on the market. This leads to a longer verification time because tests are provided at the HW level, making the finding of implementation bugs and the validation of Routing Switch functionalities more complex.

This work attempts to overcome this shortage by proposing a Verification Intellectual Property (VIP) capable of verifying the functionalities of any SpFi Routing Switch implementation and its conformance with the *Network* layer specification. The verification environment is fully compliant with the Universal Verification Methodology (UVM) standard, that is the state-of-the-art in verification technologies. All the components developed in our VIP are in fact vertically and horizontally reusable and provide full randomized stimuli generation to test the Device Under Test (DUT).

II. SpaceFibre Network Layer

The SpFi standard includes the three lowest level of the Open System Interconnection (OSI) model [9], implementing the *Network*, the Data-Link Layer (DLL), and the *Physical* layers. Actually, the DLL layer is further divided into more sub-layers, the *Multi-Lane* [10] and the *Lane* ones.

The Network Layer (NL), on which this work is focused, is responsible for the transfer of information over an SpFi network. It provides the Packet Transfer Service (PTS), that transfers SpFi packets over the network using the same format as SpW, and the Broadcast Message Service (BMS), that broadcasts short messages to all nodes on the network.

In order to connect the verification environment to the SpFi router, also some features of DLL layer must be taken into account. It provides the following services:

- *Virtual Channel* Service: supports up to 32 VCs, which send and receive Normal Characters (N-Chars) and Fill Characters (FILLs) over a SpFi link.
- *Broadcast Message Service*: sending and receiving BroadCast (BC) messages over a SpFi network.
- *Schedule Synchronisation Service*: implements QoS managing up to 32 VCs, and carrying independent flows of information over the same physical link.

The NL is responsible of transfer Packets and BC Messages over the SpFi network, passing N-Char and FILL from and to the DLL. A SpaceFibre network can be seen as a set of independent networks, called Virtual Network (VN), comprising a VC across each link which is part of the VN.

A. Packet Format and Addressing Scheme

The SpFi packet format is composed of one or more data characters followed by an End Of Packet (EOP) marker or Error End of Packet (EEP) marker. First characters represents the address, the following ones are the cargo. It is identical to the SpW packet format, meaning that a SpFi router can process both protocols at the *Network* lever, bridging them together. The interpretation of the address characters depends on the addressing scheme used to send a packet from a source node to a destination node.

Path Addressing: describes the physical port (number in 0-31 range) of the routing switch to which the packet must be forwarded.

Logical Addressing: comprises a single data character in the remaining range (32-255) at the start of the packet. A logical address represents the index entry of the Routing Table (RT), which determines the physical port the packet must be forwarded to.

Regional Logical Addressing: is a special case of logical addressing that makes it possible to have more than one logical address at the beginning of the packet, hence expanding the number of nodes accessible by a logical address.

Group Adaptive Routing: with this optional addressing scheme enabled, the logical addresses in the RT can refer to more than one physical port. Packets are routed to the first port, among those in the group, that is ready to send them to the endpoint.

Multicast: similar but mutually exclusive with Group Adaptive Routing (GAR). Packets are routed to all ports among those in the group. If one or more ports are not ready to send, all the packets are stuck.

B. Broadcast Message

The BMS is one of the most innovative feature of the SpFi standard, extending the limited Time Code (TC) mechanism of its predecessor SpW. Indeed, the BC message represents a high priority and low latency packet with respect to normal

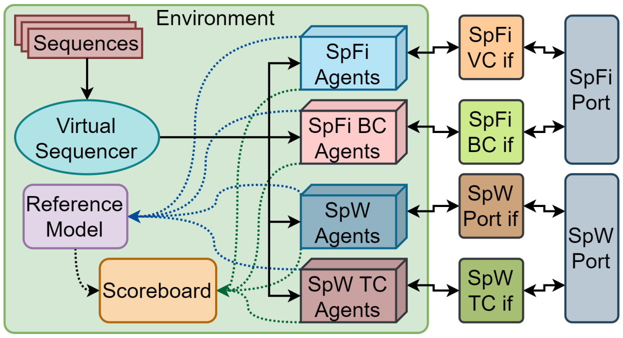


Fig. 1. The SpFi UVM Environment Connected to the SpFi Router Ports.

data. This kind of packet comprises a payload of eight bytes delimited by two SpFi control words: Start Broadcast Frame (SBF) and End Broadcast Frame (EBF). The fields in the SBF and EBF control words are needed by the router to understand the packet and verify the correctness of the message.

C. Virtual Channel Timeout

The VC Timeout is used by all VCs to detect packets that have become stuck. Each VC can have its own timeout, that must be taken into account during verification. When a packet is declared as stuck in the router, it is discarded by the input VC and an EEP marker is placed in the output VC.

III. SpaceFibre Verification Environment

The VIP presented in this work consists of a verification environment for a SpFi router written in SystemVerilog and fully compliant with the UVM. Its Transaction Level Modeling (TLM) classes allowed to work at an higher abstraction level, making possible to reuse and customize all the components of the VIP. The environment is able to validate the SpFi NL compliance of the router under test, and it is configurable with multiple ports, each of them with a generic number of VCs, to guarantee different level of QoS.

The environment must validate all the features of the SpFi router, in particular:

- *Switching Logic*: composed by the routing table, the Virtual Network Table (VNT), the BC logic, and the switching matrix.
- *Configuration Space*: composed by the router configuration, the SpFi configuration, and the SpW configuration.
- *Remote Memory Access Protocol (RMAP) Target Engine*: used to configure the memory region of the target device from remote.

A. Architecture of the Verification Environment

The environment is connected to the SpFi router through NL-level interfaces. Our environment is not dependent on the DLL implemented by the router. By using codecs [11], [12] or other router-specific protocol conversion components, it can be used with any router compliant with the SpFi NL standard. Furthermore, it is compatible with both SpFi and SpW protocols. In particular, for each SpFi VC there are:

- *SpFi TX/RX Agents*: connected to the signals of the SpFi data channels.
- *SpFi BC TX/RX Agents*: connected to the signals of the SpFi broadcast channels.

and for each SpW port (seen as a single VC by the router) there are:

- *SpW TX/RX Agents*: connected to the signals of the SpW data ports.
- *SpW TC TX/RX Agents*: connected to the signals of the SpW TC ports.

The architecture of the resulting environment is illustrated in Figure 1. Sequences are scheduled through the *virtual sequencer* of the SpFi router *environment*, which forward them to the right agent of the requested port.

In UVM, each agent is composed of a sequencer, a monitor, and a driver, as illustrated in Figure 2 for the SpFi TX agent. Sequences instruct the driver to send packets of different types, while the monitor sample the interface for packet recognition and send them to the Reference Model (RM) (compliant with the SpFi standard) to emulate the behaviour of the router.

Through the virtual sequences method implemented in our VIP, the stimuli are created and sent by the UVM test to the router. They have been designed to test the overall compliance of the DUT to the NL of the SpFi standard. There are four types of sequences:

- *Data Packet*: used to send packets from different VC, can randomize the address and the cargo.
- *Broadcast or Time Code Packet*: as the previous one, but with BC/TC-related configurations for randomization, like the BC channel, and the BC type.
- *RMAP Packet*: used to send *read*, *write*, and *read-modify-write* RMAP packets. As the others, all the fields can be randomized to generate valid sequences.
- *Read Distribution*: used to emulate the readiness behaviour of a receiving port. This makes the environment able to test various router congestion scenarios.

B. Reference Model

This work is the first in literature to present a RM emulating the NL mechanisms of a SpFi router. It interprets the packets received from the monitor of the various SpW and SpFi ports, generating the expected packets to send to the Scoreboard (SB) if needed, and updating its behaviour when new configurations are selected with RMAP packets. In our model there are a

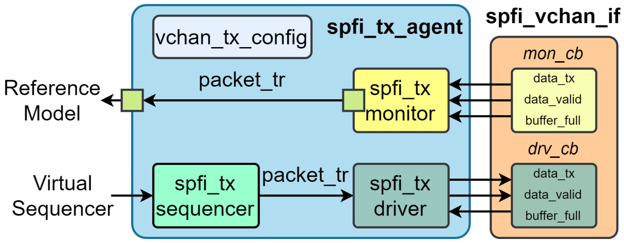


Fig. 2. The SpFi TX Agent.

couple of parallel threads for each VC and BC channel in each port of the router.

VC thread: as soon as transactions are received from SpW or SpFi agents, the RM decode them interpreting their fields, such as the destination address, the payload, the presence of an EOP marker, whether there are any other symbols after the EOP or EEP, and the FILLs. A second interpretation distinguishes data packets and RMAP packets, whose payload contains other RMAP-related fields. After that, if the packets are valid and have not been discarded for some SpFi-specific reasons, expected packets are calculated based on the internal RM configuration. The switching logic mechanism emulated by the RM is used to figure out in which $<Port, VC>$ tuple the packet is expected to arrive and send it to the SB. In case of RMAP packets, the configuration space is updated when a write or a read-modify-write command is received by the RM.

BC thread: as soon as transactions are received from SpW TC or SpFi BC agents, the broadcast logic mechanism emulated by the RM generate the correct expected packets as described in Section II-B.

C. Scoreboard

The SB is responsible for verifying whether or not a test for SpFi compliance has been passed. As the RM, it has a couple of parallel threads for each VC and BC channel in each port. In both kinds of thread, as soon as expected packets are received as transactions from the RM, they are stored waiting to be compared with the real packets arriving as transactions from the receiving agent monitors. If real packets do not find a matching expected one, they are saved as *wrong* ones. In the VC thread, in case a packet uses a GAR addressing scheme, its statistics are updated when the expected packet arrives at one of all the ports associated with that logical address. The SB removes then the generated expected ones from the other ports. For broadcast packets, since the verification environment cannot know exactly when a BC message is taken over by the router, if the BC timeout counter expires, the RM sends the expected BC and TC transactions to the SB along with an informative ”grey_zone_bc” flag. This means that the packet could have been arrived after the expiration of the timeout.

A test is considered passed by the SB if there are no wrong packets and all the received real packets have found a matching expected one.

IV. Results and Conclusions

The objective of the presented VIP is to simplify the implementation of a test plan to reach a full functional coverage of the SpFi NL standard. The verification environment permits to generate stimulus for the DUT without knowing the actual interface protocol. The user only needs to specify the kind of packet to send, data or RMAP, its size, its content, from and to which VC send it. Each unspecified field is randomized with legal or illegal values by the environment. Furthermore, the sequences can be interleaved and/or scheduled in parallel to test router congestion and timeout expiration. In summary, the set of virtual sequences designed to generate the stimuli to the DUT make the building of the test plan an easy work for the user, reducing the complexity of the verification phase, and thus saving time for product development.

A test campaign for the SpFi Routing Switch IP Core [13] developed by IngeniArs S.r.l. [14] is performed to verify the correctness of the presented verification environment and its compliance with the SpFi network-level specification. The router used during the test was configured with 4 SpFi ports, each of which has 8 VCs, and 3 SpW ports. The test plan consists of 48 different tests grouped in data packet, broadcast, and RMAP tests, plus a full random one to interleave different kind of packets.

The presented work has overcame the lack in the literature and in the market of a structured verification environment for a generic SpFi router and its conformance with Network Layer specification. The approach adopted is based on the UVM methodology, which is the state of the art in the digital design verification field, ensuring a good re-usability of all components in the environment. The provided test plan make possible to generate tests in different scenarios, understanding how the router behaves when it is used out of specification. Our verification environment can be used to identify failures at an early stage during SpFi router development and testing, thus reducing the time to market of the product.

References

[1] B. e. a. Evans, “1945-2010: 65 years of satellite history from early visions to latest missions,” *Proceedings of the IEEE*, vol. 99, no. 11, pp. 1840–1857, 2011.

[2] P. e. a. Nannipieri, *Introduction to Satellite on-Board Data-Handling*. Cham: Springer International Publishing, 5 2021, pp. 1–21.

[3] European Cooperation for Space Standardization, “SpaceFibre - Very high-speed serial link, ECSS-E-ST-50-11C,” 2019.

[4] ——, “SpaceWire - Links, nodes, routers and networks, ECSS-E-ST-50-12C Rev.1,” 2019.

[5] P. Nannipieri, L. Fanucci, and F. Siegle, “A representative spacefibre network evaluation: Features, performances and future trends,” *Acta Astronautica*, vol. 176, pp. 313–323, 2020.

[6] A. e. a. Marino, “Spaceart spacewire and spacefibre analyser real-time,” in *2020 IEEE International Workshop on Metrology for AeroSpace, MetroAeroSpace 2020 - Proceedings*, 2020, pp. 244–248.

[7] A. e. a. Leoni, “Shine: Simulator for satellite on-board high-speed networks featuring spacefibre and spacewire protocols,” *Aerospace*, vol. 6, no. 4, 2019.

[8] A. Varga and R. Hornig, “An overview of the OMNeT++ simulation environment,” in *1st International ICST Conference on Simulation Tools and Techniques for Communications, Networks and Systems*, 2008.

[9] S. Kumar, S. Dalal, and V. Dixit, “The OSI Model: Overview of the seven layers of computer networks,” *International Journal of Computer Science and Information Technology Research*, vol. 2, no. 3, pp. 461–466, 2014.

[10] G. e. a. Dinelli, “The very high-speed spacefibre multi-lane codec: Implementation and experimental performance evaluation,” *Acta Astronautica*, vol. 179, pp. 462–470, 2021.

[11] P. e. a. Nannipieri, “A serial high-speed satellite communication codec: Design and implementation of a spacefibre interface,” *Acta Astronautica*, vol. 169, pp. 206–215, 2020.

[12] G. Dinelli, P. Nannipieri, D. Davalle, and L. Fanucci, “Design of a reduced spacefibre interface: An enabling technology for low-cost spacecraft high-speed data-handling,” *Aerospace*, vol. 6, no. 9, 2019.

[13] A. Leoni, P. Nannipieri, and L. Fanucci, “Vhdl design of a spacefibre routing switch,” *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E102A, no. 5, pp. 729–731, 2019.

[14] IngeniArs S.r.l. Spacefibre router ip core. [Online]. Available: https://www.ingeniars.com/in_product/spacefibre-router-ip-core/

Networks & Protocols 2 (Long)

Coupling tools for SpaceWire on-board network: Simulaltion, Configuration and Validation

Barthélémy Attanasio
Thales Alenia Space
5 Allée des Gabians, 06150 Cannes La Bocca Cedex, FRANCE
barthelemy.attanasio@thalesaleniaspace.com

David Jameux
European Space Agency
2200 AG Noordwijk, THE NETHERLANDS
david.jameux@esa.int

Krzysztof Romanowski
ITTI
ul. Rubież 46, 61-612 Poznań
Krzysztof.Romanowski@itti.com.pl

***Abstract*— In on-board spacecraft architectures, data is increasing rapidly and SpaceWire protocol is used to define high data rate networks. SpaceWire networks have emerged in new avionics architectures for the last years and require tools for dimensioning networks, testing and validating them. One of these tools is MOST (Modelling of On-Board Spacecraft Traffic), embedding a SpaceWire block among several protocols for simulating a traffic over a network. Another one is SPACEMAN which is a discovery and configuration tool. This article explains how both tools are coupled, and how they can be used together for validation, testing or discovery.**

Keywords— SpaceWire, NDCP, RMAP, MOSTNS3, SPACEMAN, Avionics, Data Handling.

I. Introduction

SpaceWire was created specifically for space to handle high data rate. During the success of its deployment over several LSI (Large System Integrators) and space companies, several tools with different purposes have been developed. Among them, are the following:

- MOST, has been conceived to support all the phases of a spacecraft program, from design to validation tests, including error diagnostic and prediction phases. It contains specific nodes (called Building Blocks in the document) and enables modelling of many existing SpW components. Assembling them into a network allows the simulation of their real and complex behaviour to conclude with network traffic analysis.

- The SPACEMAN Network Management Tool software application can discover and configure SpaceWire networks by employing features of the NDCP protocol. SPACEMAN is also compatible with the RMAP protocol because RMAP is wildly embedded in space modules which is not necessarily the case for the NDCP protocol.

Tools are used in a way that in a program, the different phases can be followed by this two tools:

- Phase A / B1: MOST

- Phase B2 / C: MOST/ SPACEMAN

- Phase C / D: SPACEMAN

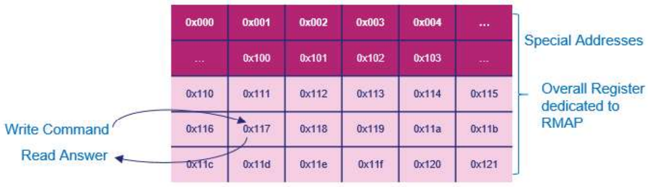
They can be coupled through NDCP or RMAP. The NDCP protocol is not part of the core simulator MOST, but there is a Building Block simulating the RMAP protocol. This Building Block was initially built to be used to simulate network traffic in terms of the amount of data transmitted, which means that the content of the packets was not entirely implemented. That is why it has been chosen to implement some registers so that SPACEMAN can discover a simulated network using RMAP commands. This development is presented hereafter.

II. RMAP Register Implementation in MOSTns3 core simulator

A. Details on RMAP Register origin

As expressed in the introduction, SPACEMAN tool needs to read information from the different nodes composing the netxwork. Using RMAP commands, SPACEMAN can discover step by step the nodes of a network. RMAP registers were not implemented in the RMAP Building Block of the MOSTns3 simulator.

Four specific registers and a generic memory space have been added, each having a real computer memory space reserved. The addresses range from 0x0 to 0x200.

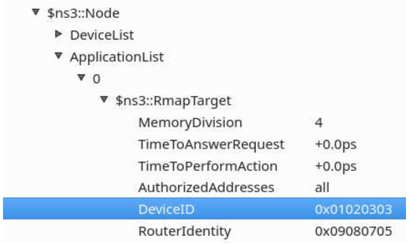


0x000	0x001	0x002	0x003	0x004	...
...	0x100	0x101	0x102	0x103	...
0x110	0x111	0x112	0x113	0x114	0x115
0x116	0x117	0x118	0x119	0x11a	0x11b
0x11c	0x11d	0x11e	0x11f	0x120	0x121

Fig. 1. Memory Management Example for RMAP registers

Two registers are configurable from the GUI by the user as the following figure illustrates:

- Device Id register: allows to identify a node
- Router Id register: allows to identify a router



```
▼ $ns3::Node
  ▶ DeviceList
  ▼ ApplicationList
    ▼ 0
      ▼ $ns3::RmapTarget
          MemoryDivision          4
          TimeToAnswerRequest     +0.0ps
          TimeToPerformAction     +0.0ps
          AuthorizedAddresses     all
          DeviceID                0x01020303
          RouterIdentity          0x09080705
```

Fig. 2. GUI representation of the RMAP registers

One register is automatically implemented depending on the device it is part of:

- Network Discovery Register:

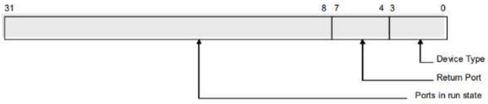


Fig. 3. Network Discovery Register details

This register is automatically updating and is essential because it gives information about the type of the device (router or unknown), the port from which the RMAP Read command arrived and the status of the device ports.

The last one is the generic memory space that can be used and configured in the simulator to write and read any additional useful information in a node.

- General Purpose Registers.

B. *Modification in SpaceWire switch architecture*

The 10XRouter Building Block of the MOST simulator is a simulated SpaceWire switch that has been modified so that in addition to its switch behavior using wormhole routing between two prots, its logical port 0 points to an RMAP application that is behaving in a proper way for a SPACEMAN discovery.

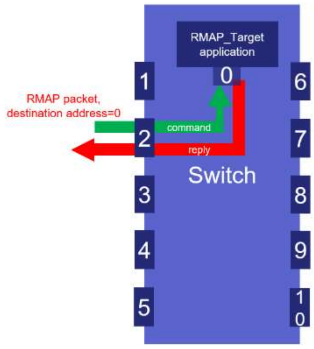


Fig. 4. 10XRouter Building Block update to handle RMAP application with new registers

That means that new RMAP application using the registers mentioned in the previous implementation is connected to this node. A validation campaign tested the behavior of the different registers depending on the RMAP commands and network configuration (number of ports connected to nodes leading to a different Network Discovery Register).

Once the registers are added to the RMAP Building Block of the simulator MOST and they are accessible using RMAP commands, a link must be created to bridge the real world (SPACEMAN tool) with the simulated network (MOST).

III. Link with HardWare Interface

To link the simulated world with the real world, a STAR-DUNDEE SpaceWire PCI Express board has been used and connected to the development PC hosting the simulator MOST. This board comprises 3 SpaceWire interfaces and can be used due to the STAR-DUNDEE libraries.

Fig. 5. STAR-DUNDEE SpW PCIe Board

By using the API, a specific Building Block in the simulator MOST has been created to interface with the board. It converts simulated packets into real SpW packets to be sent on the hardware SpW cable and vice versa.

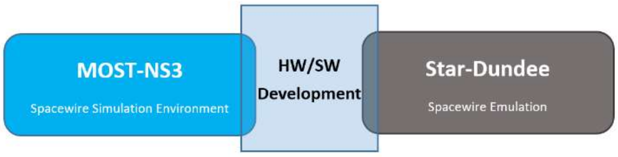


Fig. 6. HardWare SoftWare interface for MOST

The same approach is implemented with SPACEMAN to link the tool to an external interface.

IV. Coupling with SPACEMAN

Before using the new MOSTns3 features with SPACEMAN, a full validation of RMAP and switch implementation was performed to confirm the correct implementation of the overall behaviour including RMAP registers. A simple topology has been simulated and basic RMAP commands have been sent to read all registers of a target.

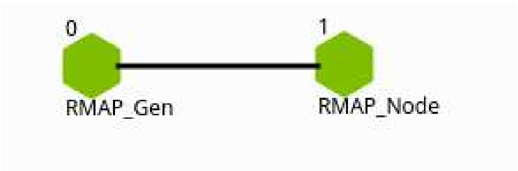
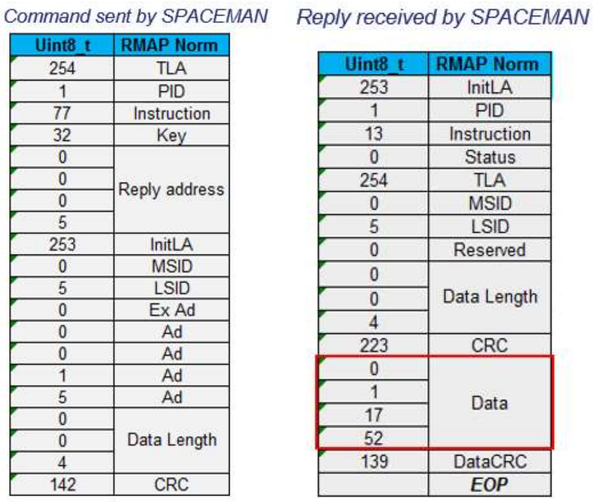


Fig. 7. Simple topology to test RMAP registers

In the previous figure, the « RMAP_Gen » node represents SPACEMAN and generates RMAP commands to read the second node "RMAP_Node" registers.



Command sent by SPACEMAN

Uint8_t	RMAP Norm
254	TLA
1	PID
77	Instruction
32	Key
0	Reply address
0	
0	
5	
253	InitLA
0	MSID
5	LSID
0	Ex Ad
0	Ad
0	Ad
1	Ad
5	Ad
0	Data Length
0	
4	
142	CRC

Reply received by SPACEMAN

Uint8_t	RMAP Norm
253	InitLA
1	PID
13	Instruction
0	Status
254	TLA
0	MSID
5	LSID
0	Reserved
0	Data Length
0	
4	
223	CRC
0	Data
1	
17	
52	
139	DataCRC
	EOP

Fig. 8. RMAP command and RMAP reply to read the DeviceID register

The details of the command and the reply are presented in the previous figure. The reply data (in red) corresponds to the DeviceID register that has been set to 0x00011137 in the MOSTGUI interface which configure the scenario of the simulation. This was validated for the several registers and the RMAP behaviour within the switch.

Then, a validation of the Hardware/Software interface was necessary. A SpaceWire cable was looped back on the PCIExpress SpaceWire card to test the new HW/SW Building Block. A sniffer was added in the middle of the SpaceWire cable. A simple simulated topology sends packets to the card and receives them back from the card.

The purpose of the sniffer is to witness the travelling of the packets through the physical network and to prove that the bridge between the software simulation and the hardware node is correct and functional:

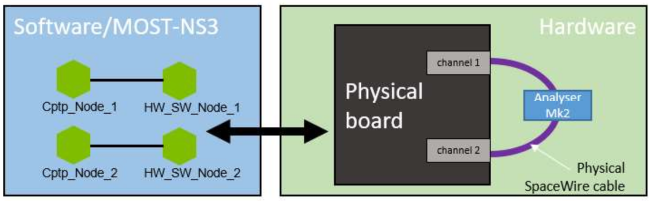
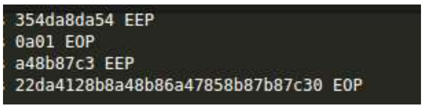


Fig. 9. Diagram of the test topology

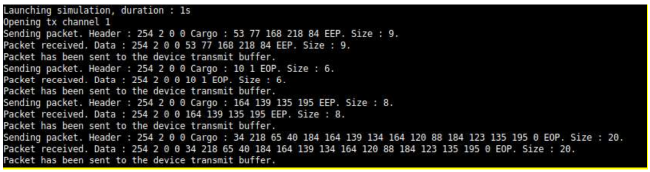
It was necessary to verify that all the NChars generated by the simulation were correctly sent to the physical board and from the physical board to the physical SpaceWire cable. For that, 4 random packets were sent by the simulated emitter node:



```
354da8da54 EEP
0a01 EOP
a48b87c3 EEP
22da4128b8a48b86a47858b87b7c30 EOP
```

Fig. 10. Fields of the random packets generated

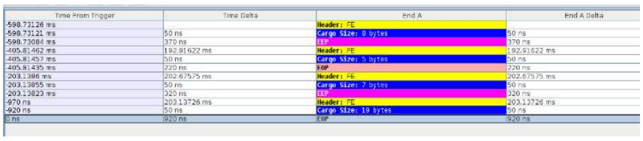
From the simulation, the following figure shows that the 4 packets have correctly been sent to the physical board with the expected fields. It also shows that all 4 packets are received with no loss of any NChars due to the loopback configuration.



```
Launching simulation, duration : 1s
Opening tx channel 1
Sending packet. Header : 254 2 0 0 Cargo : 53 77 168 218 84 EEP. Size : 9.
Packet received. Data : 254 2 0 0 53 77 168 218 84 EEP. Size : 9.
Packet has been sent to the device transmit buffer.
Sending packet. Header : 254 2 0 0 Cargo : 10 1 EOP. Size : 6.
Packet received. Data : 254 2 0 0 10 1 EOP. Size : 6.
Packet has been sent to the device transmit buffer.
Sending packet. Header : 254 2 0 0 Cargo : 164 139 135 195 EEP. Size : 8.
Packet received. Data : 254 2 0 0 164 139 135 195 EEP. Size : 8.
Packet has been sent to the device transmit buffer.
Sending packet. Header : 254 2 0 0 Cargo : 34 218 65 40 184 164 139 134 164 120 88 184 123 135 195 0 EOP. Size : 20.
Packet received. Data : 254 2 0 0 34 218 65 40 184 164 139 134 164 120 88 184 123 135 195 0 EOP. Size : 20.
Packet has been sent to the device transmit buffer.
```

Fig. 11. Traffic on the HW/SW Node

From the traffic on the sniffer (real data on the network), the analyser detected 4 packets transiting through the network, and all the packets have the size they should have:



Time From Trigger	Time Delta	End A	End A Delta
-598.73126 ms		Header: FE	
-598.73121 ms	50 ns	Cargo Size: 8 bytes	50 ns
-598.73084 ms	370 ns	EEP	370 ns
-405.81462 ms	192.91622 ms	Header: FE	192.91622 ms
-405.81457 ms	50 ns	Cargo Size: 5 bytes	50 ns
-405.81435 ms	220 ns	EOP	220 ns
-203.1386 ms	202.67575 ms	Header: FE	202.67575 ms
-203.13855 ms	50 ns	Cargo Size: 7 bytes	50 ns
-203.13823 ms	320 ns	EEP	320 ns
-970 ns	203.13726 ms	Header: FE	203.13726 ms
-920 ns	50 ns	Cargo Size: 19 bytes	50 ns
0 ns	920 ns	EOP	920 ns

Fig. 12. Packet detection in the analyzer

All of these observations validate the MOST behavior with RMAP and the link between the simulated part of MOST and the PCIExpress SpaceWire card.

V. Coupling with SPACEMAN

The last goal of this paper is to finally validate the overall link between tools by connecting the SPACEMAN software with its the SPACEMAN PCIe board and MOST with its MOST PCIe board. These two boards are connected by SpaceWire wire to link SPACEMAN and MOST.

The validation with SPACEMAN was performed by ITTI with the following topology taking into account simple examples:

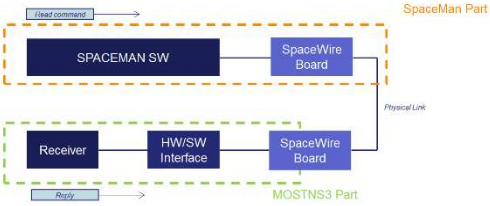


Fig. 13. ITTI validation tests

Network discovery did not require modifications in the SPACEMAN application. However, operational parameters of SPACEMAN needed adjustment (using the available set of run-time options), since the timing of command-and-reply packet transactions with a simulated network are significantly different than with a physical one, while time-out events are an important mechanism of the network discovery algorithms used by SPACEMAN.

And then a most complex case in which several nodes and switches have to be discovered:

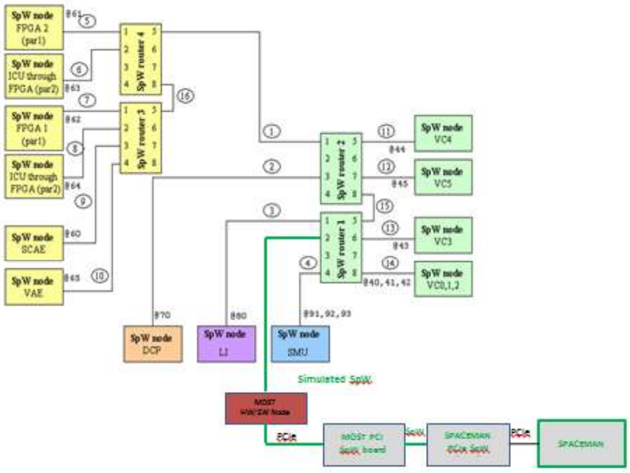


Fig. 14. SPACEMAN linked to complex Simulated Network by MOST

After having been configured in RMAP mode, SPACEMAN starts its routing for discovering the network and sends RMAP commands, step by step gathering information about the nodes composing the network. Even if not all the information are implemented in MOST, the essential ones enable to discover the network. In fact, some other secondary information could have been implemented but without a high added value for SAPCEMAN. After several commands sent and received, finally SPACEMAN discover all the nodes of the network. It is possible then for SPACEMAN to configure the network if needed.

VI. Conclusion

These two interesting SpaceWire tools that are SPACEMAN and MOST are now interoperable, they can be used in early phases as well as during Avionics Test Bench process. The flexibility and the User Interface are well suited for an easy use. For future works and evolution, it can be imagined to have the two tools on the same computer and having a single HardWare interface. It could also be possible to make a link with other simulator or SpaceWire tools.

References

[1] Short Paper, SpaceWire Network Management Using Network Discovery and Configuration Protocol, 2021

[2] ESA (European Space Agency). Standard ECSS-E-50-12CRev1, **“Space engineering. SpaceWire – Links, nodes, routers and net-works. European cooperation for space standardization**”. Noordwijk: ESA Publications Division ESTEC, 2019

[3] ESA (European Space Agency) Standard ECSS-E-50-51C **“ Space engineering. SpaceWire – SpaceWire Protocol Identification”**, Noordwijk: ESA Publications Division ESTEC, 2010. 15 p.

[4] ECSS Standard ECSS-E-ST-50-52C, **“SpaceWire - Remote memory access protocol”**, European Cooperation for Space Data Standardization, 2010

[5] ECSS Standard ECSS-E-ST-50-53C, **“SpaceWire - CCSDS packet transfer protocol”**, European Cooperation for Space Data Standardization, 2010

[6] ECSS-E-ST-50-54 Draft 1.8, **“SpaceWire Network Discovery & Configuration Protocol”**, 2016

[7] Dellandrea, B., Gouin, B., Parkes, S., & Jameux, D. (2014). “**MOST: Modeling of SpaceWire & SpaceFibre Traffic Applications and Operations: On-board Segment”**. In *European Space Agency, (Special Publication) ESA SP* (Vol. SP 725). European Space Agency

Hi-SIDE: Monitoring, Control and Test Software in a SpaceFibre Network

Dave Gibson
STAR-Dundee
Dundee, Scotland, UK
david.gibson@star-dundee.com

Balint Furdek
STAR-Dundee
Dundee, Scotland, UK
balint.furdek@star-dundee.com

Chris McClements
STAR-Dundee
Dundee, Scotland, UK
chris.mcclements@star-dundee.com

Stuart Mills
STAR-Dundee
Dundee, Scotland, UK
stuart.mills@star-dundee.com

Stephen Mudie
STAR-Dundee
Dundee, Scotland, UK
stephen.mudie@star-dundee.com

Steve Parkes
STAR-Dundee
Dundee, Scotland, UK
steve.parkes@star-dundee.com

***Abstract*—The aim of the Hi-SIDE project [1] was to develop and demonstrate technologies that enable future high-speed on-board data-handling systems. The Hi-SIDE demonstration system consisted of several elements including a SpaceFibre Camera and an instrument simulator generating high data-rate payload data; two processing elements providing compression and encryption; a PC-Based Mass-Memory providing onboard storage and playback functions; two downlink systems providing Radio Frequency (RF) and optical links; a File-Protection Scheme (FPS) to protect against errors or outages in the optical downlink; and a Control Computer used to monitor and control the other elements.**

Each of the Hi-SIDE instruments, processing, storage, and downlink elements were interconnected via SpaceFibre using the STAR-Tiger SpaceFibre [2] Routing Switch; and monitored, configured and controlled by the Control Computer.

As part of the Hi-SIDE project, software was designed and developed by STAR-Dundee to monitor and control the other elements. In addition, to demonstrate the high data-rate capabilities of the processing and downlink elements in the Hi-SIDE system, software was designed and developed by STAR-Dundee to support the transmission, storage, and playback of files encoded in Consultative Committee for Space Data Systems (CCSDS) Space Packet Protocol (SPP) [3] and Transfer Frame (TF) [4] packets at data rates of over 10 Gbit/s.

This paper describes the monitoring, control and test software that was developed by STAR-Dundee within the Hi-SIDE project and provides performance results.

Keywords—SpaceFibre, Hi-SIDE, On-Board Data-Handling

I. Introduction

The High-Speed Data-Chain (HSDC) system developed in the Hi-SIDE project demonstrates the full on-board data-chain consisting of several elements. The elements included two instruments, two processing systems, a PC-Based Mass-Memory, Radio Frequency (RF) and optical downlinks, a File Protection Scheme (FPS) used to protect against errors and outages in the optical downlink, and a Control Computer used to monitor and control the other elements. The following sections provide a brief description of these elements.

A. Instruments

The two instruments, designed and developed by STAR-Dundee, were a SpaceFibre Camera used to capture and transmit images, and an instrument simulator used to transmit files. Each instrument used two protocols to encapsulate data. Firstly, user data items e.g., image frames or files, were segmented and encoded in CCSDS SPP packets. To identify the start, middle and end of user data items, the sequence flags within the SPP packet primary header were used. Each data source in the HSDC system was assigned its own SPP Application Identifier (APID) to distinguish between different data products. Secondly, the SPP packets were encapsulated in Payload Data Encapsulation Protocol (PDEP) packets which contained the path and logical addresses, the Protocol Identifier (PID), the Source Logical Address (SLA) and a sequence number. The PDEP format was in accordance with the ECSS-E-ST-50-51C SpaceWire Protocol Identification standard [5].

A photograph of the SpaceFibre Camera connected to the SpaceFibre network is shown in Figure 1.

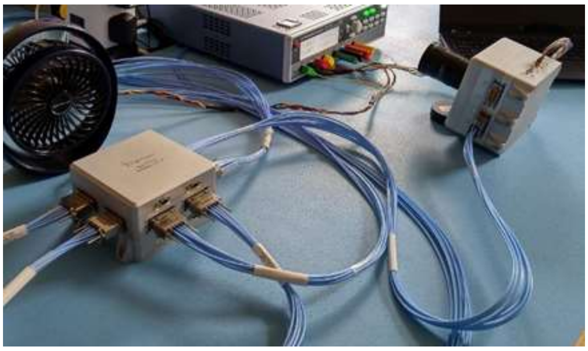


Fig. 1. SpaceFibre Camera

In Figure 1, the photograph shows the SpaceFibre Camera at the top-right connected to the STAR-Tiger SpaceFibre Routing Switch at the bottom-left using a quad-lane Elara-to-Elara SpaceFibre cable assembly. The SpaceFibre link was running at a lane rate of 6.25 Gbit/s, providing a link signalling rate of 25 Gbit/s.

The instrument simulator consisted of a STAR-Ultra PCI Express (PCIe) board [6] running on a standard desktop Personal Computer (PC) with a software application used to receive commands from the Control Computer, encapsulate files in SPP and PDEP packets and transmit them over the SpaceFibre network.

A photograph of the STAR-Ultra PCIe board is shown in Figure 2.

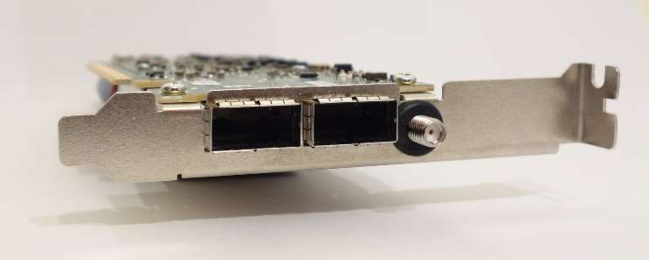


Fig. 2. STAR-Ultra PCIe

In Figure 2, the photograph shows the front-panel of the STAR-Ultra PCIe board. The board has two quad-lane SpaceFibre interfaces that use Quad Small Form-Factor Pluggable (QSFP) connectors. The SpaceFibre lanes can run at a signalling rate of 6.25 Gbit/s, providing a link signalling rate of 25 Gbit/s. In the HSDC system, the instrument simulator was connected to the STAR-Tiger SpaceFibre Routing Switch using a dual-lane QSFP-to-Elara cable assembly, providing a link signalling rate of 12.5 Gbit/s.

B. *Data Processors*

The two data processors included in the HSDC system were the Data Compression Module used to compress files encapsulated in SPP and PDEP packets, and the High-Performance Data-Processor (HPDP) used to compress or encrypt files encapsulated in SPP and PDEP packets.

The Data Compression Module [7] was produced by Airbus Defence and Space and included the Hyperspectral Compression Engine (HCE) developed by the National and Kapodistrian University of Athens (NKUA).

A photograph of the Data Compression Module connected to the SpaceFibre network is shown in Figure 3.

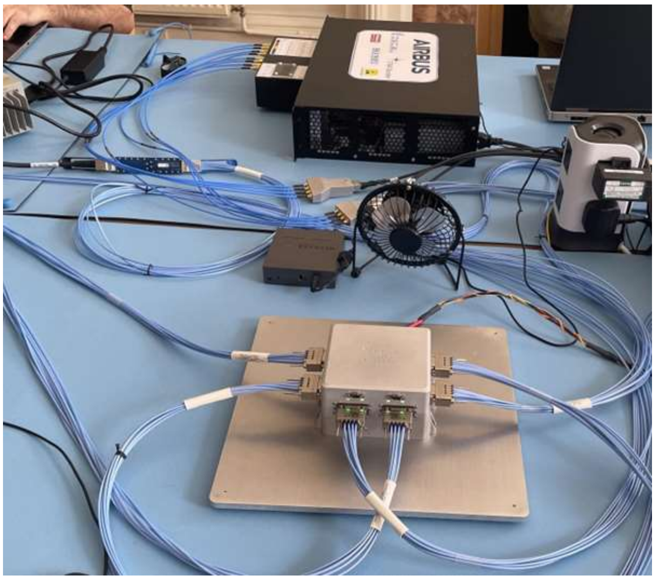


Fig. 3. Data Compression Module

In Figure 3, the photograph shows the Data Compression Module at the top connected to the STAR-Tiger SpaceFibre Routing Switch at the bottom using various cables and adaptors to convert between Sub Miniature Push-on Micro (SMPM) and Elara.

The HPDP [8] was produced by Integrated Systems Development (ISD) and included algorithms to perform data compression and encryption.

A photograph of the HPDP connected to the SpaceFibre network is shown in Figure 4.

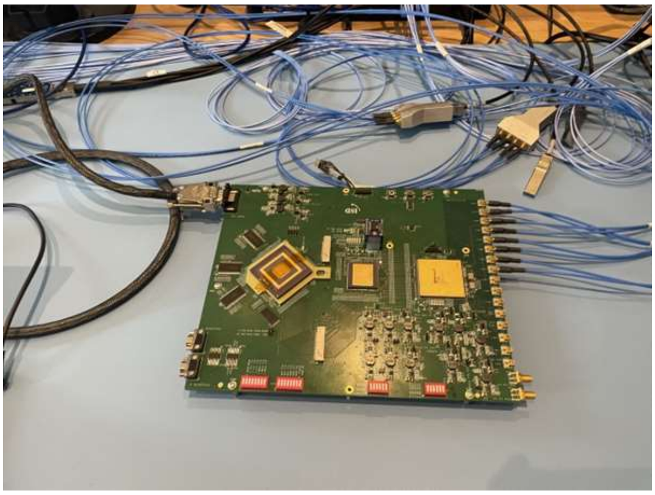


Fig. 4. High-Performance Data-Processor

In Figure 4, the photograph shows the HPDP connected to the SpaceFibre network. The HPDP was connected to the STAR-Tiger SpaceFibre Routing Switch using various cables and adaptors to convert between Sub Miniature Push-On (SMP) and Elara.

C. *PC-Based Mass-Memory*

The PC-Based Mass-Memory, designed and developed by STAR-Dundee, consisted of a STAR-Ultra PCIe board running in a standard desktop PC with a software application used to receive commands from the Control Computer; receive and store data sent by the instruments or data processors; and playback data to the data processors or downlink elements.

The PC-Based Mass-Memory and its software are described in more detail in Section III.

D. *Downlinks*

The two downlink elements included in the HSDC system were the RF Downlink used to transmit data to ground using two Ka-band transmitters, and the Optical Downlink used to transmit data to ground using a laser communication terminal.

The RF Downlink [9] was produced by TESAT and included a 2 W Ka-band Solid-State Power Amplifier (SSPA) developed by ERZIA, and a demodulator developed by Kongsberg for the ground segment.

A photograph of the RF Downlink connected to the SpaceFibre network is shown in Figure 5.

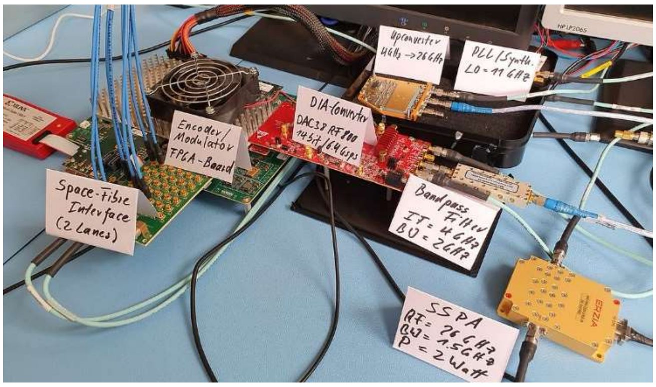

Upconverter 4GHz -> 26GHz
PLL/Synth. LO = 11 GHz
D/A-Converter DAC38RF800 14bit/6.4Gsps
Encoder/ Modulator FPGA-Board
Space-Fibre Interface (2 Lanes)
Bandpass Filter IF = 4GHz BW = 2GHz
SSPA RF = 26 GHz BW = 1.5GHz P = 2 Watt
ERZIA

Fig. 5. RF Downlink [11]

In Figure 5, the photograph shows the RF Downlink connected to the SpaceFibre network at the left. The RF Downlink was connected to the STAR Tiger Routing Switch using various cables and adaptors to convert between SMP and Elara.

The Optical Downlink [10] was produced by the German Aerospace Center (DLR) in collaboration with TESAT and included a File Protection Scheme (FPS) used to protect against errors and outages in the optical link.

E. *Control Computer*

The Control Computer, developed by STAR-Dundee, consisted of a STAR-Ultra PCIe board running in a standard desktop PC with a software application used to monitor and control the other elements.

The Control Computer and its software are described in more detail in Section IV.

II. Test Equipment Software

Early in the Hi-SIDE project, the STAR-Ultra PCIe board was designed and developed by STAR-Dundee to provide the Hi-SIDE partners with a SpaceFibre interface board to use during the design and development of their own elements.

In addition to the STAR-Ultra PCIe board, supporting test applications were designed, implemented, and provided to the Hi-SIDE partners in order to simplify integration later in the project.

The following sections describe the STAR-Ultra PCIe software and the supporting test applications.

A. *STAR-Ultra PCIe Software*

The software that was developed to support the STAR-Ultra PCIe board included:

- A high-performance PCIe driver used to transmit and receive data between the host PC and the board.
- A device configuration application used to configure the SpaceFibre lanes and links.
- A link analysis application used to support capturing and visualisation of SpaceFibre traffic.

The STAR-Ultra PCIe device driver was integrated with STAR-Dundee's STAR-System [11] software suite. This integration allowed the STAR-Ultra PCIe board to be used with the existing STAR-System applications and Application Programming Interfaces (APIs).

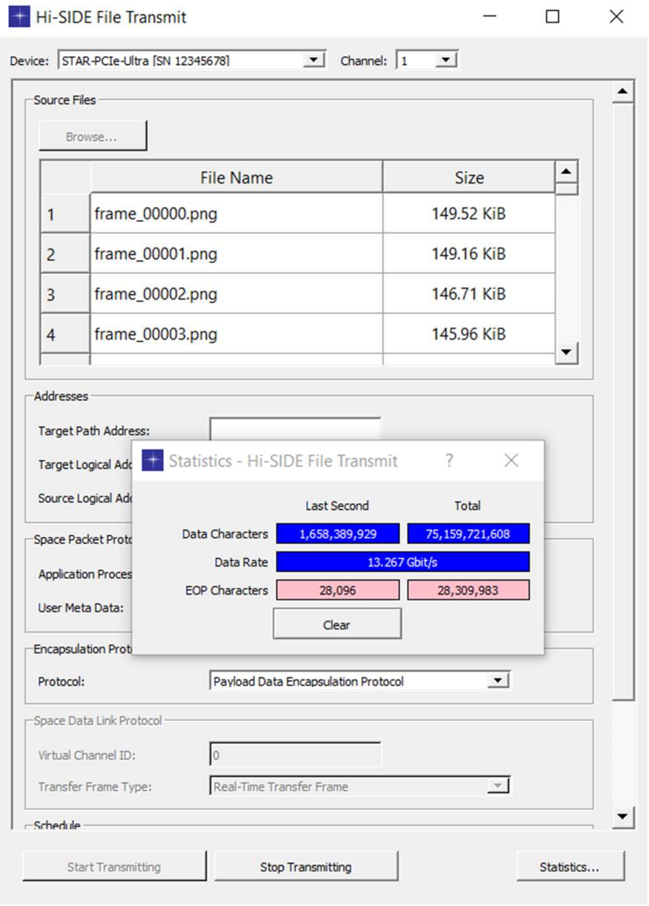
The performance objective for the STAR-Ultra PCIe was to be able to transmit and receive packets at over 10 Gbit/s in both directions simultaneously in order to support the overall 10 Gbit/s performance objective of the Hi-SIDE project. As described in Section III, this objective was achieved, for example, in applications such as the PC-Based Mass-Memory software and, as described in Section II.B, the Hi-SIDE File Transmit and File Receive applications.

B. *Test Applications*

To support the development of the Hi-SIDE elements, and to ensure that protocols were implemented consistently across the project, STAR-Dundee developed test applications used with the STAR-Ultra PCIe board.

The Hi-SIDE File Transmit application was used to encode one or more files in the protocols used in the Hi-SIDE project. For example, files could be encoded in SPP and PDEP packets for transmission to a processing element, or in SPP, TF and TFEP packets for transmission to a downlink element.

A screenshot of the Hi-SIDE File Transmit application is shown in Figure 6.


Hi-SIDE File Transmit

Device: STAR-PCIe-Ultra [SN 12345678]   Channel: 1

Source Files

	File Name	Size
1	frame_00000.png	149.52 KiB
2	frame_00001.png	149.16 KiB
3	frame_00002.png	146.71 KiB
4	frame_00003.png	145.96 KiB

Addresses

Target Path Address:

Target Logical Ad

Source Logical Ad

Space Packet Prot

Application Proces

User Meta Data:

Encapsulation Prot

Protocol: Payload Data Encapsulation Protocol

Space Data Link Protocol

Virtual Channel ID: 0

Transfer Frame Type: Real-Time Transfer Frame

Schedule

Statistics - Hi-SIDE File Transmit

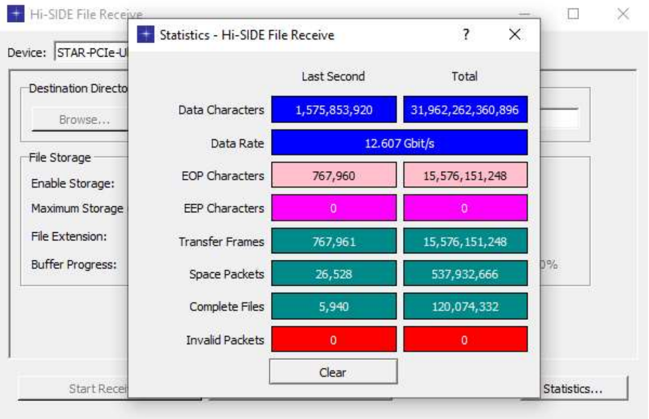
	Last Second	Total
Data Characters	1,658,389,929	75,159,721,608
Data Rate	13.267 Gbit/s	
EOP Characters	28,096	28,309,983

Fig. 6. Hi-SIDE File Transmit

In Figure 6, the screenshot shows the Hi-SIDE File Transmit application sending a list of image files encoded in SPP and PDEP packets at approximately 13 Gbit/s.

The Hi-SIDE File Receive application was used to receive, decode and optionally store files encoded in SPP and PDEP; or SPP, TF, and TFEP packets.

A screenshot of the Hi-SIDE File Receive application is shown in Figure 7.


Hi-SIDE File Receive

Device: STAR-PCIe-U

Destination Directo

File Storage

Enable Storage:

Maximum Storage

File Extension:

Buffer Progress:

Statistics - Hi-SIDE File Receive

	Last Second	Total
Data Characters	1,575,853,920	31,962,262,360,896
Data Rate	12.607 Gbit/s	
EOP Characters	767,960	15,576,151,248
EEP Characters	0	0
Transfer Frames	767,961	15,576,151,248
Space Packets	26,528	537,932,666
Complete Files	5,940	120,074,332
Invalid Packets	0	0

Fig. 7. Hi-SIDE File Receive

In Figure 7, the screenshot shows the Hi-SIDE File Receive application in operation, receiving a list of files encoded in SPP, TF and TFEP packets at approximately 12.5 Gbit/s, or over 750000 transfer frames per second.

III. PC-Based Mass-Memory Software

The PC-Based Mass-Memory was a software implementation of a block-based file system used to receive, store and playback files.

For receiving and storing files, instruments or processing elements transmit files encapsulated in SPP and PDEP packets to the PC-Based Mass-Memory.

For playing back files, the Control Computer commands the PC-Based Mass-Memory to playback a stored file to a receiver in either PDEP or TFEP encoding modes.

The PC-Based Mass-Memory software ran on a desktop PC with the following specifications:

- Intel Core i9-9900K 3.6 GHz 8-Core CPU
- 128 GB Corsair Vengeance LPX 3200 MHz DDR4
- 2 x 500 GB Samsung 970 EVO Plus M.2 NVMe SSD
- Gigabyte Z390 UD ATX Motherboard

The PC-Based Mass-Memory's STAR-Ultra PCIe board was connected to the SpaceFibre Routing Switch via a quad-lane SpaceFibre interface with eight virtual channels. The SpaceFibre lanes ran at a signalling rate of 6.25 Gbit/s, providing a link signalling rate of 25 Gbit/s

A. Storing Data

When packets were received at the PC-Based Mass-Memory, they were filtered and stored in files based on the APID in the SPP primary packet header. A diagram illustrating this process is shown in Figure 8.

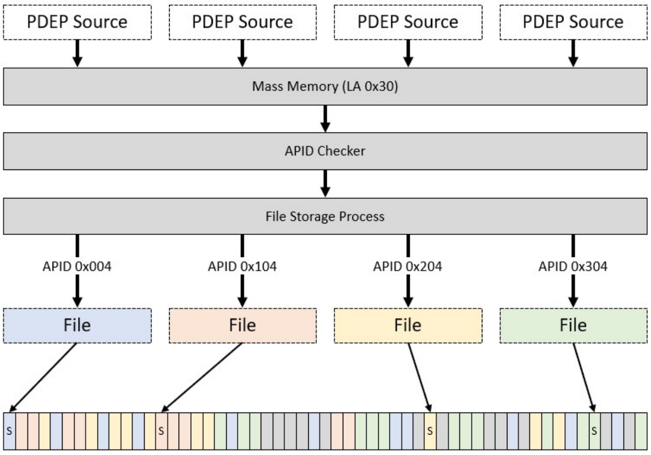


Fig. 8. PC-Based Mass-Memory

In Figure 8, there are four PDEP sources e.g., instruments or processing elements. The PDEP sources send payload data encapsulated in SPP and PDEP packets to the PC-Based Mass-Memory with logical address 0x30. When the packets are received, the PDEP and SPP packets are validated and the APID is extracted. If the APID is currently mapped to a file, the user data is passed to the file storage process. Finally, the file storage process appends the user data to the relevant file, which consists of a linked list of one or more non-contiguous blocks in the file system.

B. Playing Back Files

To playback a stored file, the Control Computer sends a playback file command to the PC-Based Mass-Memory with several parameters including the file name of the file to play back, the target logical address, the virtual channel on which to play back the file, and the requested encoding mode.

In PDEP encoding mode, the stored SPP packets are retrieved from the file system, encapsulated in PDEP headers, and transmitted to the target. In TFEP encoding mode, the stored SPP packets are retrieved from the file system, segmented and encoded in TF packets, encapsulated in TFEP headers, and transmitted to the target.

C. Control Interface

The control interface for the PC-Based Mass-Memory was a reusable software Remote Memory Access Protocol (RMAP) [12] target that was developed during the Hi-SIDE project. For example, there were commands to create files, delete files, map APIDs to files, unmap APIDs from files, playback files encoded in PDEP or TFEP encoding modes, and save and load files from local storage.

D. Performance

Prior to the demonstration, the performance of the PC-Based Mass-Memory was verified in three test scenarios: storage, playback, and simultaneous storage and playback. To measure and visualise performance, VC utilisation parameters in the STAR-Tiger SpaceFibre Routing Switch for the VCs going in and out of the PC-Based Mass-Memory were sampled by the control software and plotted over time during the tests.

A performance chart for the storage test scenario is provided in Figure 9.

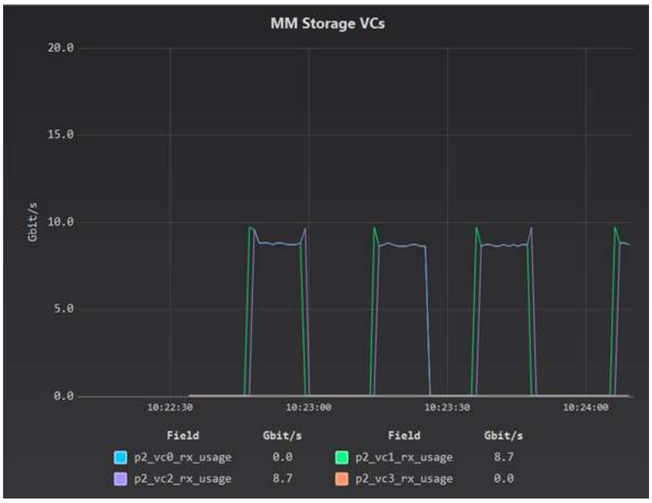


Fig. 9. Storage Performance

In Figure 9, the chart shows the data rates going into the PC-Based Mass-Memory during file storage operations from two PDEP sources sending files on VCs 1 and 2. As shown in the chart, the performance is approximately 8.5 Gbit/s on each VC, or approximately 17 Gbit/s in total on the link.

A performance chart for the playback test scenario, using PDEP encoding mode, is provided in Figure 10.

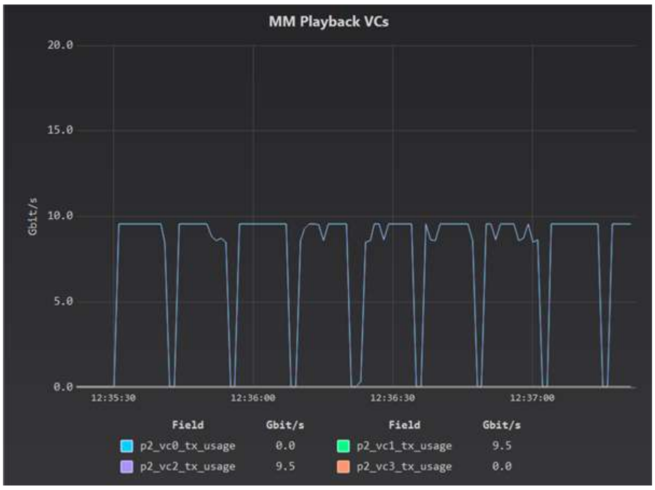


Fig. 10. Playback Performance (PDEP Mode)

In Figure 10, the chart shows the data rates coming out of the PC-Based Mass-Memory during file playback operations in PDEP encoding mode to two receivers. The playback operations are using VC 1 and VC 2 and as shown in the chart, the performance is approximately 9.5 Gbit/s on each VC, or approximately 19 Gbit/s in total on the link.

A performance chart for the playback test scenario, using TFEP encoding mode, is provided in Figure 11.

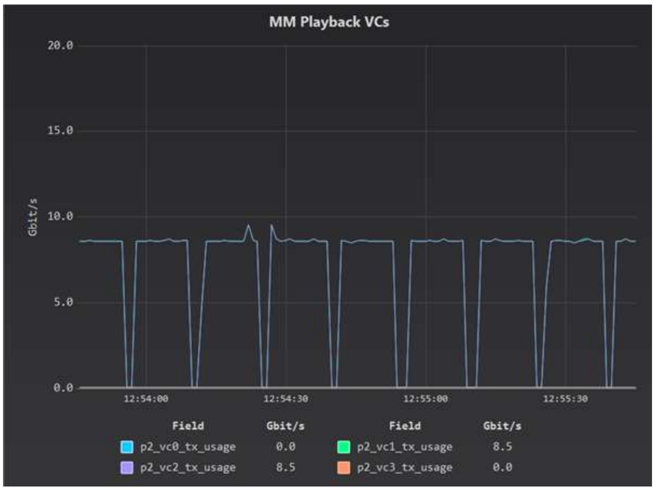


Fig. 11. Playback Performance (TFEP Mode)

In Figure 11, the chart shows the data rates coming out of the PC-Based Mass-Memory during file playback operations in TFEP encoding mode to two receivers. In this case, the performance is lower due to the additional overhead of TFEP encoding and the performance is approximately 8.5 Gbit/s on each VC, or approximately 17 Gbit/s in total on the link.

A performance chart for the simultaneous storage and playback test scenario is provided in Figure 12.

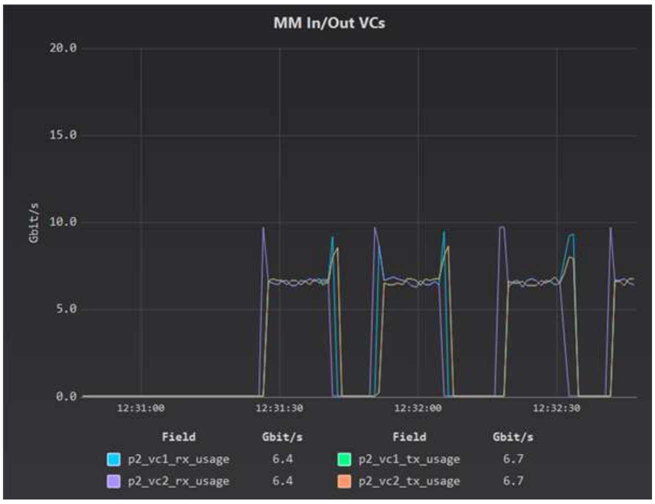


Fig. 12. Simultaneous Storage and Playback Performance

In Figure 12, the chart shows the data rates going in and out of the PC-Based Mass-Memory during four simultaneous file storage and playback operations. The four plots are showing the two VCs used to receive data for storage, and the two VCs used to playback files. As shown in the chart, the data rate is approximately 12.5 to 13 Gbit/s in each direction, or an aggregate of 25 to 26 Gbit/s.

During the execution of these tests, the control software was used to send commands to the PC-Based Mass-Memory to create the files, map APIDs to the files, and playback the stored files. It was also used to monitor and visualise the performance by sampling the VC utilisation parameters within the STAR-Tiger SpaceFibre Routing Switch and display the performance charts.

IV. Control Software

The control software, referred to as the Hi-SIDE Monitoring and Control System (MCS), ran on the Control Computer and was responsible for configuring, monitoring, and controlling the other elements during the demonstration.

Specifically, it performed the following tasks:

- Initialisation of the SpaceFibre network, including routing table configuration, setting the lane signalling rates, and link configuration.
- Initialisation of the HSDC elements, where necessary.
- Execution of control scripts used to automate the demonstration scenarios via a Python-based [13] scripting system.
- Monitoring and visualisation of relevant parameters available in the RMAP target memory of the HSDC elements.

A. *Software Layers*

A block diagram illustrating the layers of the Hi-SIDE MCS software is provided in Figure 13.

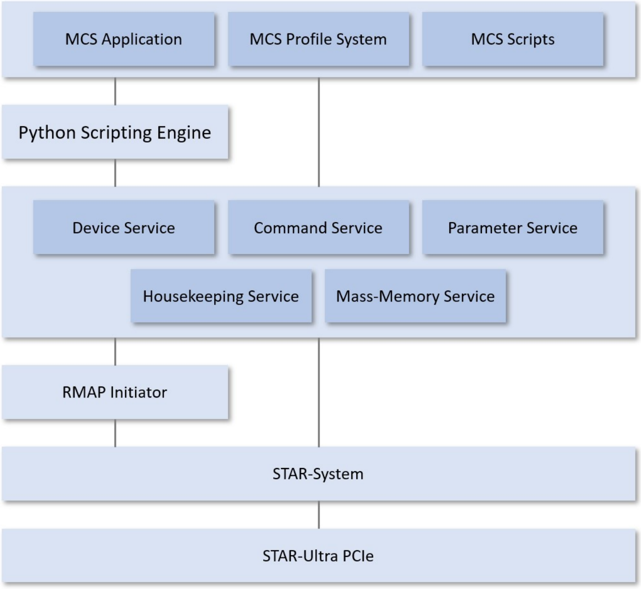


Fig. 13. Hi-SIDE MCS Layers

The top layer included the MCS Application, providing the graphical user interface; the MCS Profile System, used to describe devices in text file format; and the MCS Scripts, which were Python scripts used to automate control of the other elements.

The next layer provided the Python Scripting Engine, which included an embedded Python interpreter and Python wrappers around the services layer below.

The services layer provided the core functionality such as device management, scheduling and executing commands, parameter management, housekeeping, and interacting with the PC-Based Mass-Memory.

The RMAP Initiator was the interface to the SpaceFibre network and was used by the services to send RMAP commands to the other elements and process the corresponding RMAP replies.

Finally, the bottom two layers included STAR-System and the STAR-Ultra PCIe, as described in Section II.

B. *MCS Profiles*

The MCS Profile file format was designed to allow each HSDC element to be described electronically so that the control software was aware of how to communicate with the element, the commands available to control the element, and the parameters available to monitor the element.

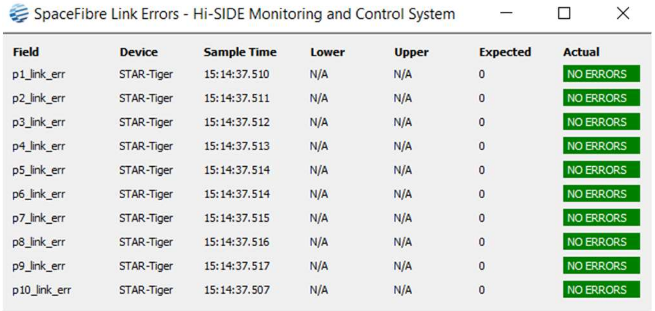
MCS Profiles were text files processed by the control software and containing the element's characteristics. Each MCS Profile contained an element's name, command path, reply path, and key, used to determine how to access an element's RMAP target memory. Following this, an element was described in terms of memory areas, registers, fields, and buffers. Each field was assigned one of the built-in types such as integer or hexadecimal, or a user-defined type created to translate between raw values and meaningful strings. In addition, optional monitoring information could be provided such as lower and upper limits or expected values.

The MCS Profiles were then processed when a project was opened to populate the library of remote devices, commands, and parameters available to the services.

C. *Parameter Tables and Charts*

After processing the MCS Profiles for each HSDC element, any parameters within an element's RMAP target memory were available to be added to the housekeeping plan, meaning they were then sampled periodically and could be displayed in parameter tables or charts.

An example of a parameter table is shown in Figure 14.


SpaceFibre Link Errors - Hi-SIDE Monitoring and Control System

Field	Device	Sample Time	Lower	Upper	Expected	Actual
p1_link_err	STAR-Tiger	15:14:37.510	N/A	N/A	0	NO ERRORS
p2_link_err	STAR-Tiger	15:14:37.511	N/A	N/A	0	NO ERRORS
p3_link_err	STAR-Tiger	15:14:37.512	N/A	N/A	0	NO ERRORS
p4_link_err	STAR-Tiger	15:14:37.513	N/A	N/A	0	NO ERRORS
p5_link_err	STAR-Tiger	15:14:37.514	N/A	N/A	0	NO ERRORS
p6_link_err	STAR-Tiger	15:14:37.514	N/A	N/A	0	NO ERRORS
p7_link_err	STAR-Tiger	15:14:37.515	N/A	N/A	0	NO ERRORS
p8_link_err	STAR-Tiger	15:14:37.516	N/A	N/A	0	NO ERRORS
p9_link_err	STAR-Tiger	15:14:37.517	N/A	N/A	0	NO ERRORS
p10_link_err	STAR-Tiger	15:14:37.507	N/A	N/A	0	NO ERRORS

Fig. 14. Parameter Table

In Figure 14, the screenshot shows a parameter table containing the link error parameters for the SpaceFibre links within the STAR-Tiger Routing Switch. For these parameters, they had an expected value of 0, so they were displayed in green when at the expected value, and red when they were not.

Multiple examples of charts were provided in Section III, as the Hi-SIDE MCS software was used during the PC-Based Mass-Memory's performance verification testing.

D. *MCS Scripts*

For the final demonstration, several Python scripts were created to automate the demonstration scenarios using the Python wrappers around the core services.

A diagram illustrating the Hi-SIDE MCS's scripting system is shown in Figure 15.

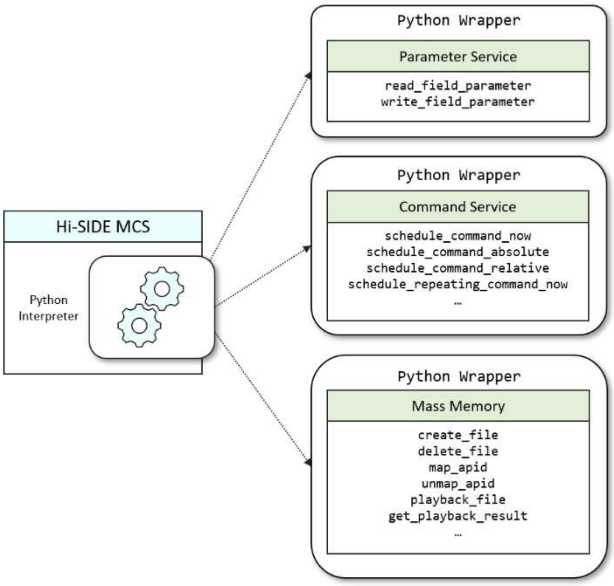


Fig. 15. MCS Scripting System

In Figure 15, the diagram shows some of the core services on the right, with each service having a Python wrapper. A Python Interpreter was then embedded in the Hi-SIDE MCS application and used to execute scripts that called the Python wrappers.

The final demonstration included the following scenarios:

- SpaceFibre Camera: Storage and Playback
- Instrument Simulator: Storage and Playback
- Compressor: Playback, Compression and Storage
- HPDP: Playback, Encryption and Storage
- RF Downlink and Demodulator: Playback
- File Protection Scheme

Scripts were created to automate the scenarios listed above, with the main script demonstrating the SpaceFibre Camera, Instrument Simulator, Data Compression Module, HPDP, and RF Downlink scenarios simultaneously.

While the demonstration scenarios were being executed, the Hi-SIDE MCS application was periodically sampling the VC utilisation registers within the STAR-Tiger Routing Switch in order to visualise the data flows going in and out of the PC-Based Mass-Memory on the various Virtual Networks.

A chart showing the traffic going to the PC-Based Mass-Memory for storage is provided in Figure 16.

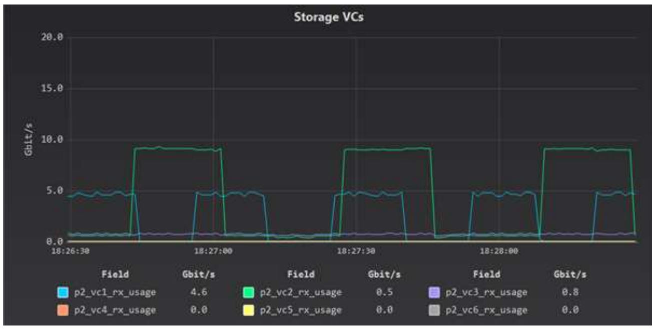


Fig. 16. Demonstration (Storage)

In Figure 16, the chart is showing data rates plotted over time for the VCs going into the PC-Based Mass-Memory. In this chart, the following data flows are shown:

- VC 1 (blue line):
  - SpaceFibre Camera sending 8 GB of images to the PC-Based Mass-Memory for storage.
- VC 2 (green line):
  - Instrument Simulator sending 16 GB of data to the PC-Based Mass-Memory for storage.
  - Compressor sending one half of the compressed data to the PC-Based Mass-Memory for storage.
- VC 3 (purple line):
  - Compressor sending the other half of the compressed data to the PC-Based Mass-Memory for storage.

When these operations overlap, the total data rate of traffic being stored in the PC-Based Mass-Memory simultaneously is over 13 Gbit/s (SpaceFibre Camera is approximately 4.5 Gbit/s, Instrument Simulator is approximately 9 Gbit/s, Compressor is approximately 0.5 Gbit/s per output stream).

A chart showing the traffic coming out of the PC-Based Mass-Memory during playback is provided in Figure 17.

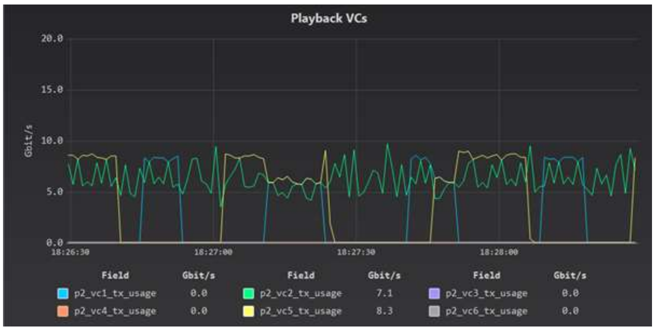


Fig. 17. Demonstration (Playback)

In Figure 17, the chart is showing data rates plotted over time for the VCs going out of the PC-Based Mass-Memory used to playback data. In this chart, the following data flows are shown:

- VC 1 (blue line):
  - Playback of 8 GB of SpaceFibre Camera images in PDEP mode to an Image Display receiver.
- VC 2 (green line):
  - Playback of uncompressed hyperspectral images in PDEP mode to the Compressor.
- VC 5 (yellow line):
  - Playback of 16 GB of Instrument Simulator data in TFEP mode to a receiver.

Due to the bursts of uncompressed hyperspectral images to the Compressor, it's more difficult to tell the total data rate of traffic when operations overlap. However, they overlap just to the left of the centre of the chart and the three playback operations are approximately 4.5 to 5.5 Gbit/s each, resulting in a total data rate of at least 13 Gbit/s.

V. Conclusions

During the Hi-SIDE project, STAR-Dundee developed various software drivers, APIs, and applications, designed to support the partners on the project, monitor and control the demonstrator, and meet the performance objectives.

At the lowest level, the STAR-Ultra PCIe board provided the interface to the SpaceFibre network, used by the partners during the development of their own elements, and in the final demonstration by the Instrument Simulator, PC-Based Mass-Memory, and Control Computer, which were developed by STAR-Dundee.

To support the STAR-Ultra PCIe board, a high-performance PCIe driver was developed for Windows and Linux operating systems and integrated with the STAR-System software suite.

On top of the STAR-Ultra PCIe driver and STAR-System, software was developed to transmit and receive files encoded in CCSDS SPP and TF protocols; store and playback files in the PC-Based Mass-Memory; and monitor and control the HSDC elements in the Control Computer.

The Hi-SIDE project culminated in the integration and demonstration of the HSDC elements, which took place successfully in June 2022 at STAR-Dundee's office in Dundee, Scotland.

ACKNOWLEDGEMENT

The Hi-SIDE project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 776151.

REFERENCES

[1] Hi-SIDE Consortium, https://www.hi-side.space/.

[2] ECSS Standard ECSS-E-ST-50-11C, "SpaceFibre – Very High-Speed Serial Link", Issue 1, European Cooperation for Space Data Standardization, May 2019, available from http://www.ecss.nl.

[3] CCSDS Standard 133.0-B-2, "Space Packet Protocol", Blue Book, Consultative Committee for Space Data Systems, June 2020, available from https://public.ccsds.org/.

[4] CCSDS Standard 732.0-B-4, "AOS Space Data Link Protocol", Blue Book, Consultative Committee for Space Data Systems, October 2021, available from https://public.ccsds.org/.

[5] ECSS Standard ECSS-E-ST-50-51C, "SpaceWire Protocol Identification", Issue 1, European Cooperation for Space Data Standardization, 5 February 2010, available from http://www.ecss.nl.

[6] STAR-Dundee, "STAR-Ultra PCIe", https://www.star-dundee.com/products/star-ultra-pcie/

[7] Hi-SIDE Consortium, "Data Compression Module", https://www.hi-side.space/hi-side

[8] Hi-SIDE Consortium, "High-Performance Data Processor Payload Unit", https://www.hi-side.space/copy-of-hi-side-optical-data-link

[9] Hi-SIDE Consortium, "High Rate RF Downlink Transmitter", https://www.hi-side.space/hi-side-rf-data-link

[10] Hi-SIDE Consortium, "Optical Data Link", https://www.hi-side.space/hi-side-optical-data-link

[11] S. Mills and S. Parkes, "A Software Suite for Testing SpaceWire Devices and Networks", Proceedings of Data Systems in Aerospace (DASIA) Conference, Barcelona, Spain, 2015.

[12] ECSS Standard ECSS-E-ST-50-52C, "SpaceWire – Remote Memory Access Protocol", Issue 1, European Cooperation for Space Data Standardization, 5 February 2010, available from http://www.ecss.nl.

[13] Python Software Foundation, "Python", https://www.python.org/

Missions & Applications (Short)

(Approved for Public Release, Distribution Unlimited).

DESIGN AND QUALIFICATION OF TWELVE METER SPACEWIRE FLEXIBLE PCB BACKPLANE FOR A SPACE ROBOTICS SERVICER

Derek Schierlmann
Naval Center for Space Technology
Naval Laboratory Code 8243
Washington, D.C., USA
derek.schierlmann@nrl.navy.mil

Eric Rossland
Naval Center for Space Technology
Naval Laboratory Code 8243
Washington, D.C., USA
eric.rossland@nrl.navy.mil

William Vincent
Naval Center for Space Technology
Naval Laboratory Code 8243
Washington, D.C., USA
william.s.vincent@nrl.navy.mil

***Abstract*— NRL researchers recently designed and built a SpaceWire link over a flexi-cable which is more than three times as lossy (in decibels per meter) as a cable constructed to section 5.2 of the SpaceWire specification. The length and lossy nature of this flex-cable made closing a SpaceWirelink at our requirement of 100Mbps exceptionally difficult. Significant testing and analysis were undertaken before settling on an implementation that uses LVDM drivers in place of the LVDS called out in the SpaceWire specification in order to close a link at up to 200Mbps. This paper describes the advantages SpaceWire brings to the RSGS payload and the significant analysis, simulation, and testing needed to design and qualify a SpaceWire link to transmit data at up to 200 Mbps over this twelve-meter flex-cable.**

I. Introduction

The Defense Advanced Research Projects Agency's (DARPA's) Robotic Servicing of Geosynchronous Satellites (RSGS) robotic servicing vehicle, consisting of a robotic payload developed by the Naval Research Laboratory (NRL) and a space vehicle bus developed by a commercial partner, intends to demonstrate a dexterous robotic operational capability in Geosynchronous Orbit that can both provide increased resilience for the current US space infrastructure and be the first concrete step toward a transformed space architecture within the next five years. To accomplish this mission, the RSGS robotic payload integrates data from and sends commands to situational awareness cameras, inspection cameras, remote command and telemetry boxes, rendezvous / proximity operation sensors, payload control computers and the spacecraft bus. SpaceWire was an ideal choice to enable these components to communicate in a robust and reliable way because it is:

- Supported by detailed specification(s) which enables simple interface control documents (ICDs) and interface discussions
- Widely adopted with implementations available on many flight hardware cards, devices and boxes
- Easy to implement and adopt for those who do not already have a SpaceWire interface
- Supported with significant GSE available off the shelf
- Flexible to accommodate work arounds, changes and customizations such as those covered in this paper

Critical to the RSGS payload are two seven-degree-of-freedom, two-meter-long robotic arms. The robotic arms are slender and dexterous and make use of arm-mounted cameras to reach into tight spaces on the customer spacecraft without the risk of unintentional contact. The robotic arms host a capable electronics suite of sensors and provide the ability to host smart tools at the arm tip. SpaceWire provides a robust and flexible mechanism that allows RSGS to multiplex tool data, sensor data and high-speed camera data and transmit that to the payload control computers. To prevent snagging on appendages, our robotic arms implemented a cabling solution making use of flexible printed circuit board technology to create a twelve meter 'flex-cable' to connect the end of arm electronics with the rest of the payload electronics. Flex-cables have significant flight heritage, including on multiple Mars rover missions, and by using flexible circuit cables the RSGS arm can pass nearly a thousand individual electrical signals through a 2" x.250" space held rigidly near the arm structure. RSGS uses a twelve meter 'flex-cable' to connect the end of arm electronics with the rest of the payload

Distribution Statement A: Approved for public release. Distribution is unlimited electronics. This flexibility doesn’t come without a price as we found out in initial testing.

II. Two problems

During initial integration of the cameras to the arm, two problems were uncovered that impacted SpaceWire. The first was that we were unable to establish high rate digital communications on arm and the second was that once communication was established, the link failed to pass data at any speed when the tool drive motor was activated. Both issues needed to be solved to enable reliable communications on the arm but this paper will focus only on the first, establishing high rate communications.

To maintain maximum dexterity, each joint in the arm requires service loops in the flexible cable which adds to the overall length. So, despite the arm being approximately two meters in length, electrical signals travel through twelve meters of cable to reach from the base of the arm to the tip. The slenderness enabled by the arm’s flex-cable means that the conductors which carry camera and other end of arm data are thirty times smaller and thus have significantly higher resistance when compared to a typical 26 AWG round wire harness. In initial testing, eye diagrams captured of S100 ieee-1394a transmission on arm show that while 100Mbps data transmission was possible at those rates, it violated the specification and was a negative margin situation which would likely get worse as typical space environmental factors further degrade the signal. Initial designs used ieee-1394 based cameras due to previous NRL investigations into rad-hard ieee-1394a chipsets [1] but ultimately the SpaceWire advantages won out and the mission converted to SpaceWire. SpaceWire testing has similar results, but less granularity since the test setup was limited via software to only 10Mbps and 200Mbps. Transmission worked at 10Mbps but 200Mbps immediately failed.

III. What was happening?

When we tested the flexi-cable, we found that while its impedance is within specifications for both ieee-1394 and SpaceWire (at 100ohms +/- 6), the measured cable attenuation is above specification: flex cable attenuation is 12dB (-75%) at 100MHz which is 9.8 dB more than the 2.2dB allowed in the 1394a specification. This out of spec attenuation gets worse as frequency increases: at 200MHz the cable attenuates 18dB vs 3.2dB in the spec (14.8 dB over) and at 400MHz, the flex attenuates 25dB vs 5.8dB in the spec (19.2dB over). Analysis in Johnson and Graham [2] which is supported by interpretation of the time domain reflectometry test results on the arm (figure 1) suggest that for this length and at these bit rates, attenuation is dominated by skin effect losses in the copper [3]. This analysis also eliminated dielectric losses as a contributing factor and focused our efforts towards adding copper as the primary way to reduce attenuation. More copper is needed, but adding thickness to the flex was not an option because a larger flex would cause a mechanical redesign of the arm. For similar reasons, changing to other cabling or cable materials were not options. In summation, the design space was: get more copper on each trace without increasing the overall thickness of the cable or changing the basic flex trace layout.

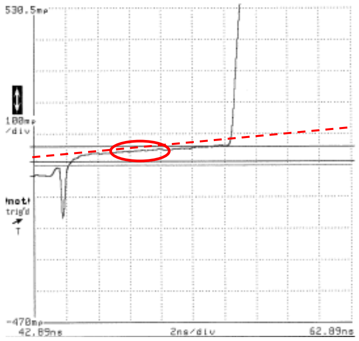


Figure 1: A time domain reflectometry trace of the flexible printed circuit board cables in the first iteration of the RSGS robotic arm design. Note that while the impedance is within spec for ieee-1394 and SpaceWire, the lack of connector impedance discontinuity (a) and the positive and increasing slope of the trace (b). Both of these speak to a significant loss of TDR signal resolution specifically from cable attenuation.

NRL engineers built a model of the flex cable in Hyperlynx which allowed the team to quickly try numerous digital protocols, speeds, drivers, receivers, and termination configurations to see which had a chance to work in this lossy cable. We simulated and tested a host of different driver-receiver combinations, including other protocols and other paradigms such as repeaters inline. Through these simulations, the team settled on the use of an LVDM driver / receiver (Aeroflex UT54LVDM031/32LV) and the recommended 35 Ohm termination resistor.

Oscilloscope traces were taken of data transmission on the robotic arm to compare and extend results of these simulations. Simulation correlation testing provided the real world data needed to answer a host of questions including comparison of flex to 41' of Gore 26AWG cable (figure 2); LVDS vs. LVDM; max rate testing; termination resistance options; clock recovery techniques; GSE configuration; physical trace variations (e.g. other data pairs); data source (simulated v. real SpaceWire) and data pattern testing (FCT-NULL v. random). In all, we collected data in 207 different configurations of on arm communications. When tested on the arm, 100Mbs LVDM SpaceWire transmission was reliable with margin and transmission at 180Mbps was possible, but nothing over 150Mbps had margin, and 200Mbs was still not achieved.

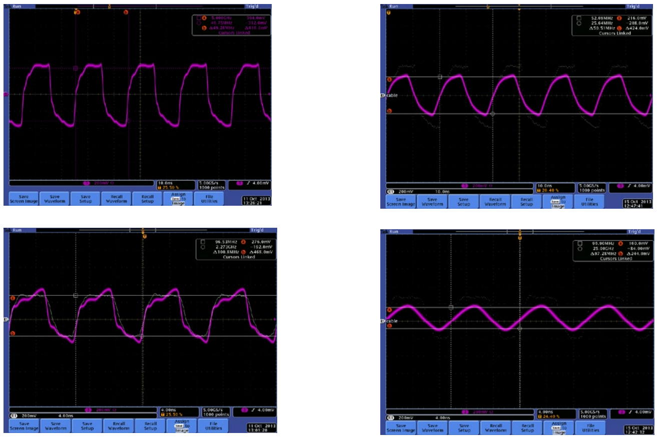


Figure 2: Comparing data transmission at the same rate, by the same driver and receiver over 26AWG round wire cable (1a, 2a) to that take over the same length of flexible PCB cable shows the significantly higher attenuation of the flexible PCB cable. LVDM driving (channel d2) with pigtails into 35Ω. Vertical scale = 200mV/div for all waveforms; vertical cursors for most screens; horizontal scale changes.

In an effort to get more margin back, we next undertook some testing to confirm the cause of the high loss and try alternate PCB trace layout configurations. We fabricated a flex cable for thermal and life testing and populated it with three trace configurations that we could use to characterize the flex loss. The configurations were chosen from table 6.2 in Johnson and Graham [2] and represented differing configurations what could be achieved in the design space of the current flex. The options included a slight improvement to the original (stripline) design, a more aggressive option (microstrip) and an extreme option intended to produce minimum attenuation (also microstrip). The stripline was effectively the original trace configuration, but loosely coupled, vs. tightly coupled as on the original design. When tested, the data transmission results confirmed that we were skin effect limited and the maximum achievable data rate tracked conductor cross section area. Thus, the microstrips significantly outperformed the stripline in max speed but the stripline geometry outperformed microstrip geometries in noise rejection. Additionally, the test validated that loosely coupled differential traces decrease resistive losses in that the stripline configuration achieved the same maximum speed as the original design despite being smaller (eight mils wide vs. nine mils wide originally). In general, the results directly followed the predictions in Johnson and Graham [2]. Exiting this testing the design rules were established: use loosely coupled 50 Ohm striplines of maximum size that fit in the flex space requirements.

V. Flex Redesign: Small Changes for Big Impact

Significant effort was spent trying to remove as many error sources from the flex as possible so that length would be the dominant aspect. We maximized conductor area to the extent we could, but the main improvements were in ground planes and routing consistency, as highlighted below. We added 2 layers to the flex without increasing overall size which added an outer chassis layer(s) to the original inner shields. The inner grounds covered the traces with overlap on the sides to minimize EMI except for the connector interface. We kept the grounds as one trace across the flex rather than separate to minimize EMI. And at the flex to connector interface, pins were reserved for top and bottom internal ground planes to be made available which kept coverage to all except for the solder pads. Additionally, we maintained continuity of the ground plane connection from round wire harness (or next flex section) to the flex ground plane so that the currents could flow from both ground planes correctly. We also spent effort on routing consistency such as spreading the signals out to minimize cross-talk, shooting for 5:1 spacing to height ratio per literature [2]. We also adjusted the flex routing (bends) and adjusted trace lengths as needed to make the two traces in each pair exact duplicates as well as for the pair of signals in each transmit/receive pair. The results of all these changes we now had fewer impedance discontinuities, better matched signal pairs and more consistent of conductor geometries.

VI. Data

Using the test procedure developed in house [4], we were able to consistently test the current performance of the SpaceWire link through the build of the arm and payload. Immediately, our improvements were validated and we quickly got 200Mbps LVDM transmission. Our results were looking so good that we reintroduced LVDS to the design to help with configuration management. We used LVDM at the end of arm electronics and everything else in the network was standard LVDS SpaceWire. The performance kept improving as we replaced GSE with flight components, but after component level, we stopped taking eye diagrams (figure 3) so we had no ability to see quality of transmission at these rates in later configurations. All test results at were executed ambient temperature and pressure, but we have done testing which shows minimal impact at predicted operational temperatures. In the end, we are able to achieve full rates even with the presence of noise.

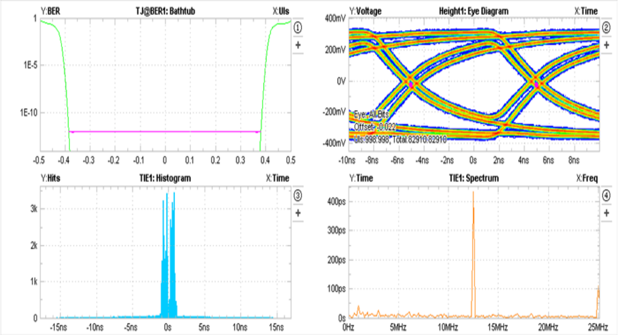


Figure 3: The last eye diagram taken shows 100Mbps data transmission with margin over EM flex cable driven by EM and GSE hardware. Feature of note (a) clean eye. (b) Wide bathtub suggests <1E-12 BER likely. (c) TIE histogram is non-gaussian with two peaks.

TABLE I.

SpaceWire Max Data Rate Test Results History				
	Max Data Rate			
Link	**Best Link**		**Worst Link**	
Signal Type	**LVDM**	**LVDS**	**LVDM**	**LVDS**
Initial testing with or without Arm motion***	100	10	-	-
Initial testing with TD running***	0	0	-	-
Simulation correlation testing	180	Not tested	-	-
Test Flexes, P1 (Stripline) configuration	180	Not tested	-	-
Test Flexes,P1 (Stripline) with noise	160	Not tested	-	-
Test Flexes, P1 (Stripline) with noise and filtering	180	Not tested	-	-
EM Flex on Bench (at NRL)	200	70	200	-

RAA configuration, arm off (at Vendor, no filter)	200	70	200	70
…with Servos on, brakes engaged	200	70	70	90
…with Servos, brakes engaged + TD running	200	70	0	0
RAS configuration (at NRL, with filter)	200	100	200	100
…with Servos on, brakes engaged	200	100	200	100
…with Servos on, brakes disengaged	200	100	200	100
…with Servos, brakes disengaged + TD running	200	100	200	100
Notes: white row = no noise blue row = with noise *** = passed S100 firewire ~LVDM, passed 10Mbps SpaceWire, but no other SpaceWire rates tested due to SW driver issue				

VII. CONCLUSIONS

The key things that are team took away from this testing started with an appreciation of SpaceWire. In addition to the advantages listed above, it was surprisingly tolerant of this level of violation of the specification. With good engineering discipline and attention to detail, we were successful at recovering margin, we were able to give margin away to ease I&T and System Engineering problems: such as the re-introduction of LVDS to the design and an additional two meters of off arm cabling. From our experiences, the little things make a difference. The largest improvement likely was use of LVDM, but attention to detail on the trace geometry (loosely v. tightly coupled trace pairs) and obsessive care in layout (trace matching and consistency) bought nearly as much margin. We can offer three recommendations to our audience. First, LVDM is a great option if struggling with a high loss transmission media. Second, Our test procedure for validating and verifying margins on a SpaceWire link was quite useful. Lastly, When in doubt, trust your literature. Specifically, High Speed Signal Propagation: Advanced Black Magic by Johnson and Graham when tested, correctly predicted all test trends and often predicted the results.

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA).

REFERENCES

[1] K. D. Wolfram and H. J. Bloom, "New radiation-hardened high-speed serial data bus for satellite onboard communication," IGARSS 2004. 2004 IEEE International Geoscience and Remote Sensing Symposium, 2004, pp. 159, doi: 10.1109/IGARSS.2004.1368969.

[2] H. Johnson, G. Martin, High-Speed Signal Propagation: Advanced Black Magic, New Jersey: Prentice Hall PTR, 2003

[3] TDR Impedance Measurements: A Foundation for Signal Integrity. Tektronix application note 55W_14601_2. Available online https://download.tek.com/document/55W_14601_2.pdf

[4] D. Schierlmann, M. Long, S. Bagnall, Qualification Of One And Two Interconnects In A Spacewire Link," Proceedings of the International SpaceWire Conference 2022.

[5] D. Schierlmann, P. Jaffe, "SpaceWire Cabling in an Operationally Responsive Space Environment," Proceedings of the International SpaceWire Conference 2007.

[6] D. Schierlmann, E. Rossland, P. Jaffe, "Lessons Learned From Implementing Non Standard Spacewire Cabling For TACSAT-4," Proceedings of the International SpaceWire Conference 2008.

[7] P. Jaffe, G. Clifford, J. Summers, "SpaceWire for Operationally Responsive Space as part of TacSat-4," Proceedings of the International SpaceWire Conference 2007.

[8] Pioneer Circuits web page https://www.pioneercircuits.com/pci-advantage/

[9] H. Johnson, G. Martin, High-Speed Digital Design: A Handbook of Black Magic, New Jersey: Prentice Hall PTR, 1993

[10] C. R. Paul, Introduction to Electromagnetic Compatibility, John Wiley & Sons, 1992.

Evaluation of the SpaceWire-D network and architecture through HAYABUSA2 optical navigation system

Missions and Applications, Short paper

Hiroki Hihara
Space Engineering Division
NEC Space Technologies, Ltd.
Fuchu, Tokyo, Japan
hihara@nec.com

Junpei Sano
Space Engineering Division
NEC Space Technologies, Ltd.
Fuchu, Tokyo, Japan
j-sano-ti@nec.com

Takeshi Oshima
Space Systems Division
NEC Corporation
Fuchi, Tokyo, Japan
t-ohshima@nec.com

Tatsuaki Okada
Institute of Space and Astronautical Science (ISAS)
Japan Aerospace Exploration Agency (JAXA)
Sagamihara, Kanagawa, Japan
okada@planeta.sci.isas.jaxa.jp

Naoko Ogawa
Institute of Space and Astronautical Science (ISAS)
Japan Aerospace Exploration Agency (JAXA)
Sagamihara, Kanagawa, Japan
ogawa.naoko@jaxa.jp

Yuichi Tsuda
Institute of Space and Astronautical Science (ISAS)
Japan Aerospace Exploration Agency (JAXA)
Sagamihara, Kanagawa, Japan
tsuda.yuichi@jaxa.jp

***Abstract*—We developed a hard real-time optical navigation system based on SpaceWire-D draft standard for HAYABUSA2 asteroid probe. HAYABUSA2 accomplished its primary mission successfully in the vicinity of asteroid Ryugu in November 2019. It brought back samples of Ryugu on December 6th, 2020 to the Earth. Digital Electronics and Optical Navigation Camera (DE-ONC) was used for scientific observation as well as for optical navigation using real-time image recognition. Thanks to the integrated architecture referring to SpaceWire-D we were able to analyze deterministic onboard networks to fulfill the timeliness and latency requirement. Image tracking functions were successfully performed during autonomous and automatic touch-down onto the surface of Ryugu. No feature points were missed, and they were tracked continuously within limited latency using radiation hardened devices. We report the evaluation result of the design approach in this paper.**

Keywords—SpaceWire-D, image recognition, real-time, optical navigation, artificial intelligence

I. Introduction

We developed a hard real-time optical navigation system based on SpaceWire-D draft standard [1] for HAYABUSA2 asteroid probe. HAYABUSA2 was launched in December 2014 and aimed at sample-return from a near-Earth asteroid 162173 Ryugu [2]. Its front view is shown in Fig. 1. It accomplished its primary mission successfully in the vicinity of asteroid Ryugu in November 2019. It brought back samples of Ryugu on December 6th, 2020 to the Earth.

Digital Electronics and Optical Navigation Camera (DE-ONC) was developed for scientific observation as well as for optical navigation using real-time image recognition [3, 4, 5, 6]. We developed a deterministic onboard network to fulfill the timeliness and latency requirement. SpaceWire is used both for interconnections among onboard equipment and for intra-unit connections between distributed heterogeneous processing elements (PEs). Thanks to the integrated architecture using SpaceWire-D, heterogeneous PEs worked concurrently. Virtual buses and time-slots management controlled by an initiator were key technical issues. They are proposed and are specified in SpaceWire-D draft standard. Latency was verified over wire harnesses and a backplane in a unit in accordance with the determinacy definitions of SpaceWire-D. Feature points tracking functions were successfully performed during autonomous and automatic touch-down onto the surface of Ryugu. No feature points were missed during real-time optical navigation, and they were tracked continuously within limited latency [7].

Since the central processing unit (CPU) of DE-ONC consists of heterogeneous PEs using a conventional micro-processor unit (MPU) and an Field Programmable Gate Array (FPGA), we employed middle-out approach [8, 9] to encapsulate heterogeneous PEs for programmable automatic and autonomous image recognition. Functions implemented on the heterogeneous PEs were integrated by a dedicated programming language, and the whole system can be programmed through one user programming interface [10]. We referred to the middle-out approach of parallel inference machines of the Fifth Generation Computer System, which was developed by the Institute for New Generation Computer Technology (ICOT) [8].

Fig. 1. HAYABUSA2 front view https://jda.jaxa.jp/result.php?lang=j&id=521b93ae47393f8eeefcc1aa4a29b796

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

II. HAYABUSA2 ONBOARD IMAGE PROCESSING

A. *Onboard computer for optical navigation system*

Two DE-ONCs were developed to satisfy the requirement of reduced mass, low power consumption and small size due to resource restrictions [3, 4, 5, 6]. Fig. 2 shows one DE-ONC unit, and the specifications of the DE-ONC is shown in table 1. Since we developed fast lossless and lossy image compression algorithm [10, 11], we implemented these functions by software on an MPU. On the other hand, we developed a dedicated image processing processor using an FPGA, because signal compensation and image recognition functions had to be processed in wire-rate through camera interfaces. High reliability is required for deep space probes, and hybrid and reconfigurable computing [12] such as the combination of a radiation hardened MPU and an FPGA is a typical configuration. Latency requirements of these processors are different during optical navigation operations of HAYABUSA2 mission. Therefore, we adopted SpaceWire-D draft specification [1] to satisfy and verify the latency requirements.

Fig. 2. DE-ONC

TABLE I. DE-ONC SPECIFICATIONS

Item	Specifications
Image recognition rate	2 Hz (max), 1 Hz (nominal)
Conventional processor	HR5000S (JAXA authorized MPU)
Image processing processor	Implemented on Microchip RTAX2000S
SpaceWire port	Telemetry / Command: 2 ch (redundant) Data recorder I/F: 1 ch Sensor I/F: 2 ch (nominal)
Conventional I/F	PIM (Peripheral Interface Module): 1 ch UART (RS422): 3 ch (including dedicated I/F)
Memory buffer	SDRAM: 1 Gbytes (*) Flash memory: 2 Gbytes (*) (*) Reed-Solomon encoded symbols are included.
Size (mm)	97.7 (W) × 231.3 (D) × 177.8 (H)
Mass	2.7 kg
Power consumption	14.1 W

B. *An active back plane*

We extended the deterministic communication scheme with SpaceWire and Remote Memory Access Protocol (RMAP) to inter-module communications inside a DE-ONC unit of HAYABUSA2. SpaceWire active backplane using one-chip SpaceWire router LSI (Large Scale Integration) was adopted to guarantee a hard real-time performance required for the optical navigation subsystem [6]. The one-chip SpaceWire router was developed using JAXA authorized Silicon-on-Insulator (SOI) application specific integrated circuit (ASIC). Circuit card assemblies were connected through bus connectors embedded inside the metal frames of the assemblies. Physical backplanes were eliminated, and small and low mass profiles which was required for a deep space probe was realized.

A real-time image recognition module using the image processing processor is connected to the router chip using SpaceWire and RMAP protocol. Feature point extraction results with labels are transferred through a SpaceWire the router chip to a CPU card for higher level image processing functions such as clustering and template matching. Large memory buffers are mounted on the image recognition module, and high resolution images are captured and stored into the buffers simultaneously with image recognition operation. These processing cycles are not always synchronized with the other functions such as telemetry and command processing. Virtual buses in accordance with SpaceWire-D draft standard corresponds to the bus configuration, which is explained later.

III. REFERENCING SPACEWIRE-D DRAFT STANDARD

Deterministic extension of SpaceWire has been proposed by the University of Dundee [1]. The specification enables SpaceWire networks to be used for payload and control applications using existing SpaceWire equipment. Definitions for latency to be referred to test specifications are provided. Implementation scheme of bus configurations are shown. Distinctive features to meet latency requirements are virtual buses, time slots, schedules, and error detection. Testability is augmented by these features. The correspondence with these mechanisms of the internal SpaceWire network of DE-ONC is described in this section.

A. *Virtual buses*

Two DE-ONCs are used as an optical navigation camera electronics and a digital electronics for sensor signal processing. They are called ONC-E and DE, respectively. Fig. 3 shows the block diagram of ONC-E. SpaceWire interconnections inside the unit are also shown.

The image recognition module has memory buffers to store high resolution images. The image operations are processed simultaneously with navigation calculations processed by an Attitude and Orbit Control Processor (AOCP). Note that the processing cycles of the navigation calculations do not always synchronized with the cycles of telemetry and command (T&C) processing. An RMAP initiator implemented as a software on an HR5000S MPU controls these two bus management scheme for optical navigation and T&C control using a SpaceWire router.

Virtual buses described in SpaceWire-D draft specification corresponds to the configurations. Network topology is divided into two virtual buses where all traffic is controlled by a single initiator. It allows a single SpaceWire

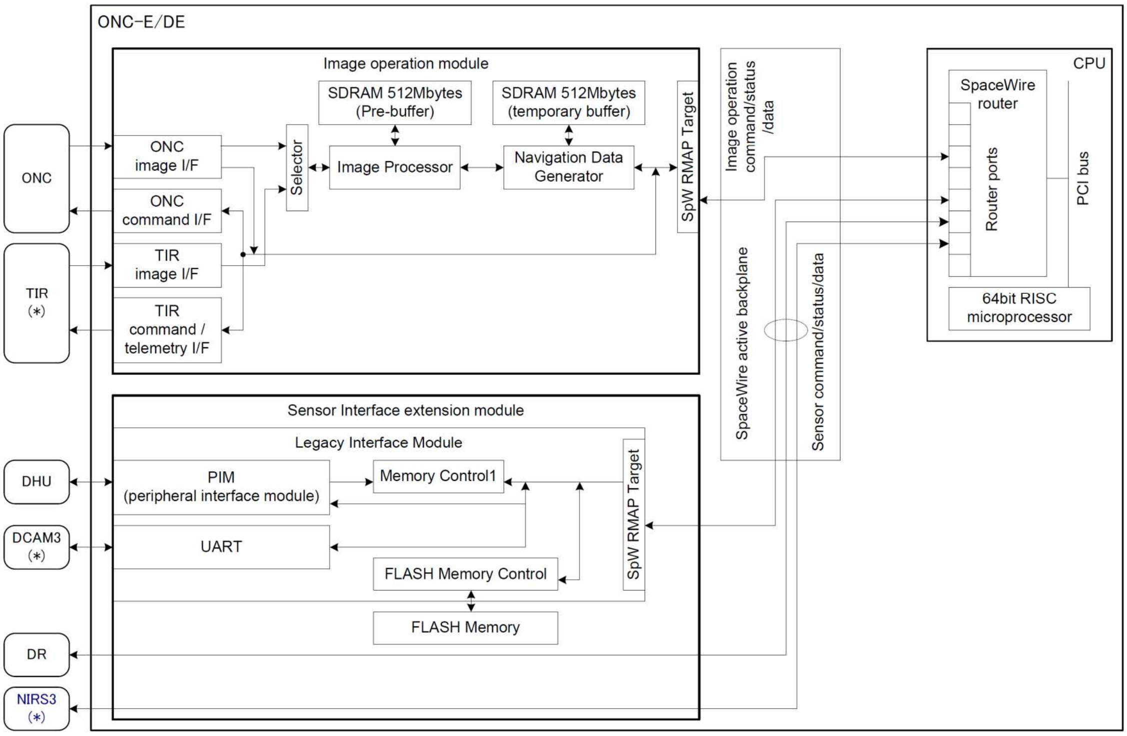
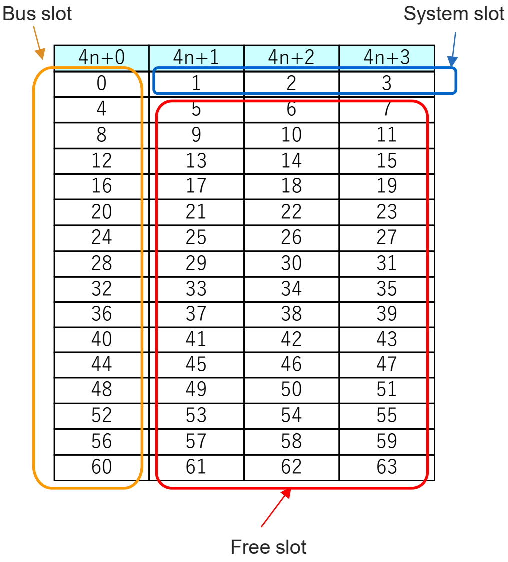

ONC−E/DE
Image operation module
SDRAM 512Mbytes (Pre-buffer)
SDRAM 512Mbytes (temporary buffer)
ONC
ONC image I/F
ONC command I/F
Selector
Image Processor
Navigation Data Generator
SpW RMAP Target
TIR (*)
TIR image I/F
TIR command / telemetry I/F
Image operation command/status /data
SpaceWire active backplane
Sensor command/status/data
CPU
SpaceWire router
Router ports
PCI bus
64bit RISC microprocessor
Sensor Interface extension module
Legacy Interface Module
DHU
PIM (peripheral interface module)
Memory Control1
DCAM3 (*)
UART
FLASH Memory Control
SpW RMAP Target
FLASH Memory
DR
NIRS3 (*)

Fig. 3. SpaceWire active backplane inside ONC-E. (*) TIR, DCAM3 and NIRS3 are connected to DE.

network to be used for payload and control applications. Existing SpaceWire equipment is used with a SpaceWire-D software layer. Two virtual buses are independent as for memory buffer resources, and the independent virtual buses can operate at the same time without blocking.

B. *Time slots*

Network time is divided into time-slots controlled by SpaceWire time-codes. One second is divided into 64 time as shown in Fig. 4. SpaceWire Time-Code corresponds to each time slot. Time Indicator (TI) is delivered as a 32-bit value by Data Handling Subsystem (DHS) for values exceeding 1 second. TI is concatenated with the 6-bit value of SpaceWire time-codes. This method was established by the activities of SpaceWire User's Group Japan [13] and based on what was established as ASTRO-H SpaceWire network design standard [14]. Scheduling virtual buses into time-slots can remove blocking, resulting in predictable RMAP execution times [15].


Bus slot
System slot
Free slot

4n+0	4n+1	4n+2	4n+3
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63

Fig. 4. Time-Slots of DE-ONC in one second.

C. *Schedules*

Two types of bus services are specified in SpaceWire-D [1]. They are Static Bus Service and Dynamic Bus Service. Static Bus Service initiates a transaction group to read and write to RMAP targets in a specific time-slot or multi-slot. Dynamic Bus Service initiates a transaction group to read and write to RMAP targets in the next available time-slot or multi-slot allocated to the dynamic bus.

Bus slots and system slots shown in Fig. 4 correspond to Static Bus Service. Transaction sequences are defined in system design before onboard operations. On the other hand, free slots shown in Fig. 4 is used in accordance with onboard sensor operation plans. These time-slots can be used for Dynamic Bus Service.

Two more bus services are specified in SpaceWire-D [1]. They are Asynchronous Bus Service and Packet Channel Service. Asynchronous Bus Service asynchronously queues and initiates individual transactions to read and write to an RMAP target sending those transactions in a group within slots allocated to the Asynchronous Bus. Packet Channel Service sends and receives application packets to and from a target by segmenting the packets so that they can be transferred using a transaction that will fit into a single slot, and sending those transactions in a group within slots allocated to the Packet Bus [15]. Similar implementations to these bus services was employed on DE-ONC.

D. *Error Detection and Testability*

Error detection scheme is shown in SpaceWire-D draft specification [1]. A network manager handles report list of errors to network manager at the end of each schedule epoch

[1]. Fault detection, isolation, and reconfiguration (FDIR) functions were implemented referring to the specification.

Scheduling virtual buses into time-slots can remove blocking, resulting in predictable RMAP execution times [15], and how to specify latency requirement is shown in SpaceWire-D. SpaceWire is used over an active backplane and external wire harnesses, and latency can be analyzed over wire harnesses and a backplane inside a unit. We used a verification system developed by JAXA and Nagoya University [16] to analyze latency.

IV. Programmability of Heterogeneous Processors

A. *Middle-out approach*

DE-ONC has heterogeneous processing elements (PEs) including a dedicated processor implemented on an FPGA. There is a wide semantic gap between an application program and PEs. And yet programmability is required even in the vicinity an asteroid over 3 hundred millions kilometers away. This is lessons learned through the experience of the first HAYABUSA asteroid probe against unknown risks. In addition to that, concurrency among the PEs has to be maintained.

Middle-out approaches were proposed and demonstrated to fill the semantic gaps [8, 9]. Concurrency was also realized with GHC and KL1 developed by ICOT [8]. We implemented a dedicated programming language to encapsulate heterogeneous PEs and to fill the semantic gaps. It is called an observation program, or Kansoku Program in Japanese. We call it Kan-pro in short [10]. Operation program sequences on DE-ONCs were optimized successfully in accordance with observed environment data around Ryugu by operators who were not the expert of heterogeneous PEs.

B. *Onboard demonstration*

Three types of optical navigation were implemented for HAYABUSA2 [7]. They are Asteroid Image Tracking (AIT), Target Marker Tracking (TMT) and Characteristic Geography Tracking (CGT). AIT aims to process images of entire asteroids. TMT outputs the center value of multiple bright spots in the image, and it consists of Normal Bright object Tracking (NBT) sub-mode and Differential Bright object Tracking (DBT) sub-mode. The size and brightness of the bright spot are evaluated in NBT, and the difference of images when a flashlight is on and off is evaluated in DBT. In CGT, feature points are tracked by correlation processing with the template image specified from an AOCP, and the representative coordinates are output.

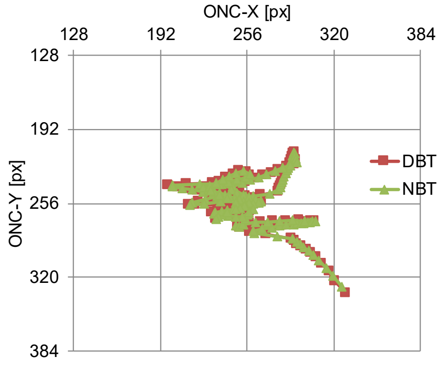


Fig. 5. A result of TM tracking in the touchdown rehearsal operation (TD1-R3) on October 25, 2018. (X - Y plot) [7]

Fig. 5 shows examples of the evaluation results using DBT and NBT. TMT operations were carried out every 4 seconds. These results show the confirmation of bright spot coordinates output through TMT. The target marker was traced using NBT and DBT alternately, and the output coordinates were almost identical. The bright spots were detected continuously and were not missed.

References

[1] S. Parkes, A. Ferrer, D. Gibson, SpaceWire-D Standard Draft D, Issue 0.15, 2014.

[2] Y. Tsuda, M. Yoshikawa, M. Abe, H. Minamino, S. Nakazawa, "System design of the Hayabusa 2 - Asteroid sample return mission to 1999 JU3," Acta Astronautica, Vol. 91, pp. 356-362, October-November 2013, doi:10.1016/j.actaastro.2013.06.028.

[3] H. Otake, T. Okada, R. Funase, H. Hihara, J. Sano, K. Iwase, S. Kawakami, J. Takada, T. Masuda, "Onboard infrared signal processing system for asteroid sample return mission HAYABUSA2," Proc. SPIE 9219, Infrared Remote Sensing and Instrumentation XXII, 921904, September 2014, doi.org/10.1117/12.2062818.

[4] T. Okada, T. Fukuhara, S. Tanaka, M. Taguchi, T. Imamura, T. Arai, H. Senshu, Y. Ogawa, H. Demura, K. Kitazato, R. Nakamura, T. Kouyama, T. Sekiguchi, S. Hasegawa, T. Matsunaga, T. Wada, J. Takita, N. Sakatani, Y. Horikawa, K. Endo, J. Helbert *et al.*, "Thermal Infrared Imaging Experiments of C-Type Asteroid 162173 Ryugu on Hayabusa2, " Space Science Reviews, 208, pp. 255-286, 2017.

[5] H. Hihara, K. Iwase, J. Sano, H. Otake, T. Okada, R. Funase, R. Kashikawa, I. Higashino, T. Masuda, "SpaceWire-based thermal-infrared imager system for asteroid sample return mission HAYABUSA2," J. of Applied Remote Sensing, 8(1), 084987, 2014, doi: 10.1117/1.JRS.8.084987.

[6] H. Hihara, K. Moritani, T. Masuda, R. Funase, H. Otake, T. Okada, "Intelligent Navigation System with SpaceWire for Asteroid Sample Return Mission HAYABUSA2," Proc. Int'l. SpaceWire Conf. 2013, pp. 308-311, 2013.

[7] N. Ogawa, F. Terui, Y. Mimasu, K. Yoshikawa, G. Ono, S. Yasuda, K. Matsushima, T. Masuda, H. Hihara, J. Sano, T. Matsuhisa, S. Danno, M. Yamada, Y. Yokota, Y. Takei, T. Saiki, Y. Tsuda, "Image-based autonomous navigation of Hayabusa2 using artificial landmarks: The design and brief in-flight results of the first landing on asteroid Ryugu," Astrodyn. 2020, 4(2): 89-103. doi: 10.1007/s42064-020-0070-0.

[8] K. Ueda, T. Chikayama, "Design of the Kernel Language for the Parallel Inference Machine," The Computer Journal, Vol. 33, No. 6, pp. 494-500, 1990, doi: 10.1093/comjnl/33.6.494.

[9] K. H. Bennett, M. P. Ward, "Theory and Practice of Middle-Out Programming to support Program Understanding," Proc. 1994 IEEE 3rd Workshop on Program Comprehension- WPC '94, pp. 168-175, 1994, doi: 10.1109/WPC.1994.341267.

[10] H. Hihara, J. Sano , J. Takada, T. Masuda, T. Okada, N. Ogawa, H. Ootake, Y. Tsuda, "Onboard signal processing system of HAYABUSA2," Proc. SPIE. 11502, Infrared Remote Sensing and Instrumentation XXVIII, 115020L, 2020, doi: 10.1117/12.2570424.

[11] J. Takada, S. Senda, H. Hihara, M. Hamai, T. Oshima, S. Hagino, M. Suzuki, S. Ichikawa, "A fast progressive lossless image compression method for space and satellite images," Proc. 2007 IEEE IGARSS, pp. 479-481, 2007, doi: 10.1109/IGARSS.2007.4422835.

[12] A. D. George, C. M. Wilson, "Onboard Processing With Hybrid and Reconfigurable Computing on Small Satellite," Proc. IEEE, Vol. 106, No. 3, pp. 458-470, March 2018, doi: 10.1109/JPROC.2018.2802438.

[13] SpaceWire User's Group, Japan, "SpaceWire Network Design Guideline", Version 1.0, 13 May 2010.

[14] T. Yuasa, T. Takahashi, M. Ozaki, M. Kokubun, M. Nomachi, H. Hihara, K. Mizushima, T. Kominato, K. Omagari, K. Masukawa, "A Deterministic SpaceWire Network Onboard the ASTRO-H Space X-Ray Observatory", Proc. Int'l. SpaceWire Conf. pp. 348-351, 2011.

[15] D. Gibson, S. Parkes, "SpaceWire-D," SpaceWire WG 24, ESA/ESTEC, 2015.

[16] M. Takada, H. Takada, Y. Chen, T. Yuasa, T. Takahashi, M. Nomachi, "Development of software platform supporting a protocol for guaranteeing the real-time property of SpaceWire," Proc. Int'l. SpaceWire Conf., pp. 80-87, 2013.

SpaceWire network and communication technology in the MMX system

Mission and Applications, Short Paper

Akira Chiba, Isao Odagi, Shinya Hirakuri, Takahiro Owaki,
Shimpei Ogino, Kazuhiro Nishikawa
Mitsubishi Electric Corporation (MELCO)
Kamakura Works
Kamimachiya, Kamakura, Kanagawa 247-8520, Japan
Chiba.Akira@db.mitsubishielectric.co.jp

Masanobu Ozaki
Institute of Space and Astronautical Science (ISAS)
Japan Aerospace Exploration Agency (JAXA)
Sagamihara, Kanagawa, 252-5210, Japan
ozaki.masanobu@jaxa.jp

***Abstract*—Mitsubishi Electric Corporation (MELCO) has applied SpaceWire for internal component links and mission instruments interface with high-speed links so far. This time MELCO had a chance to apply it for the MMX (Martian Moons eXploration) system network. This paper describes the network topology, protocols, communication technologies and lessons learned in this system. The MMX mission is a project planned launch in the mid-2020s, to explore the two moons of Mars. Data handling (DH) subsystem including two components: SMU (Satellite Management Unit) and MDP (Mission Data Processor) is responsible for the master and relay of the network.**

Thirty SpaceWire nodes and six SpaceWire Routers (in RTG4 and GR740) are in the MMX system. We apply SpaceWire as internal SMU/MDP links, interface between SMU and MDP, SMU/MDP and mission instruments, SMU and Retrievable Data Recorder (RDR). These communication data is used for not only DH but also Guidance Navigation and Control (GNC) subsystem. Although each node has variety of data size and collection frequency, software in SMU/MDP controls transaction with 64Hz timing slots to achieve acquiring all HK telemetries and observation data.

The MMX system consists of three modules (Propulsion Module, Exploration Module and Return Module) which is separated in operation. We apply SpaceWire at the interface between SMU and MDP, SMU and RDR, which interface will be separated. Therefore, we have designed the circuits in SMU to protect from external noise because these will be exposed outer space due to the separation.

RDR is the mission component and there is severe limitation of the number of harnesses between SMU and RDR interface. For this reason, we have developed Strobe-less communication technology, which is useful for the systems that have limitation on the number of harnesses.

Keywords—SpaceWire, Networks, Spacecraft Electronics

I. INTRODUCTION

Our development of SpaceWire interface began with communication for an internal component. We has expanded to inter-component, then to within spacecraft systems. For internal component, SpaceWire backplane was implemented in Satellite Control Platform (SCP), enabling up to 3Gbps [1]. In addition to low-speed communication between boards, this technology can also handle large-volume, high-speed data, such as record and replay data.

In our development of SpaceWire interface, RDDP was adopted to guarantee high reliability of high-speed, large-volume data (~100 Mbps) from sensors. PTP is applied for telemetry and telecommand interface with SCP, and RMAP for internal component [2]. The network topology is a composite configuration of mesh and spoke and hub. The mesh topology consists of control board. They can be configured as a triple-majority configuration or as a redundant system with cross-straps.

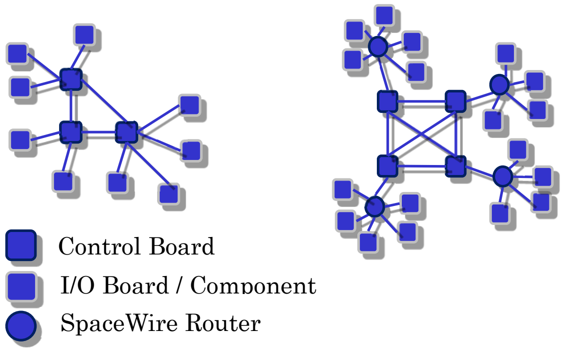


Fig. 1. Our previous network topology

A. *MMX Mission Scope*

Martian Moons eXploration (MMX) is a project to reveal the origin and evolutionary process of the Mars and moons [3][4]. Through the development and operation of the MMX system, the project aims to acquire as follows.

- Technology for the return trip between the Mars orbit and Earth
- Advanced sampling technology from the surface.
- Communication technology for deep space exploration.

MELCO is in charge of design and manufacturing of the spacecraft system, as well as support for operations until returns to Earth.

B. *Modules*

MMX consists of three modules: Propulsion module, Exploration module, and Return module. These will be detached when no longer needed to reduce mass and save propellant. Each module has following role.

- Propulsion module has an essential role before reaching and is detached near Mars.
- Exploration module has an role to survey two moons: Phobos and Deimos. It will be separated when the exploration will complete.
- Return module plays a necessary role for all mission life time.

The module configuration is shown in Fig. 2.

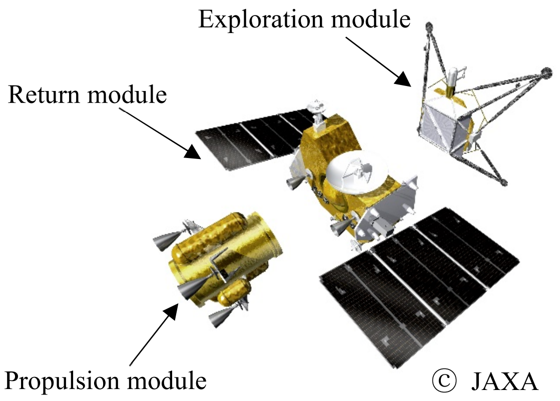


Fig. 2 MMX module configuration

II. MMX Network overview

A. Toplology

Fig. 3 shows the SpaceWire network topology for MMX. SMU and MDP have the main role of the network.

Inside SMU, there are I/F boards which have interfaces with GNC (Guidance Navigation and Control) and SM (Spacecraft Management) instruments. SpaceWire is applied between these boards and Control board. MDP performs image processing for landing. MDP transfers the results to SMU via SpaceWire. Thus, SpaceWire is applied not only for DH data, but also for GNC and SM data communication.

There are thirty SpaceWire nodes in the MMX system, with link speeds from 10 to 180 Mbps. Since the image processing results processed by MDP are used by SMU S/W, MDP and SMU must be synchronized to improve landing accuracy. Moreover, in order to communicate with many components, time slots are applied. We also use SpaceWire time-code to guarantee delay time and high reliability of communication.

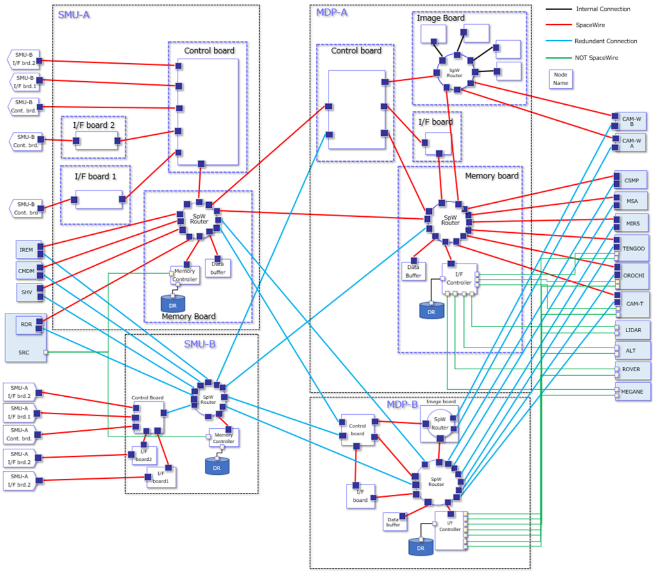


Fig. 3. Network Topology

B. Data Handling Components

We also have developed DH components: SMU, MDP and RDR. SMU acts as the brain in the spacecraft. Main functions are as follows:

- CCSDS TC/AOS
- Distributing telecommand
- Collecting, recording and replaying telemetry and observation data
- Time management
- Guidance Navigation and Control
- Spacecraft Management

Main functions of MDP are as follows:

- Distributing telecommand
- Collecting recording and replaying telemetry and observation data
- Image processing for landing and descending
- Data processing

RDR is contained in Sampling Return Capsule (SRC), which will separate from Return module near Earth and go back to surface. RDR has very large storage, which can be used for various purpose recording. RDR will bring extra scientific data back to Earth that could not be downlinked to the ground due to RF line conditions. TABLE I. shows the main specifications of each component. Fig.4 shows the block diagram of SMU and MDP.

TABLE I. SPECIFICATIONS OF SMU,MDP AND RDR

	SMU	MDP	RDR
CPU	HR5000S	HR5000S, GR740	None
Size (Typ.)	231x315x223 mm	171x315x223 mm	110×110×25 mm
Mass (Typ.)	10.5 kg	7.4 kg	301 g
Power (Max.)	67 W	74 W	6.5 W
HW Redundancy	yes	yes	no
Recording Capacity	32 GByte	32 GByte	1 TByte
Main I/O Interface	SpaceWire MIL-STD-1553B RS422	SpaceWire RS422 LVDS	SpaceWire (Strobe-less)

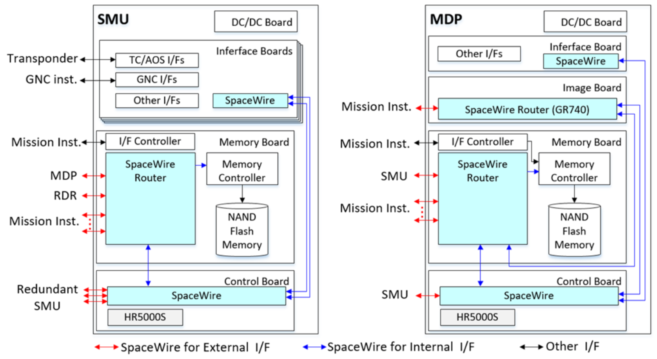


Fig. 4. Block diagram of SMU and MDP

C. Timing Scheduling

In order to collect telemetry and send telecommands in real time within one second, time slot management with time-code 64 Hz is applied. The value of time-code delivered is called a slot. S/W of SMU and MDP use this slot value to determine the data to be communicated. SW collects observation data with RMAP read. When large size of data is returned to SW, a lot of processing time is spent. To solve this problem, Initiator Logical Address (ILA) in RMAP read command is set as memory controller. This method let RMAP read reply directly input to it via SpaceWire routing (Fig.5).

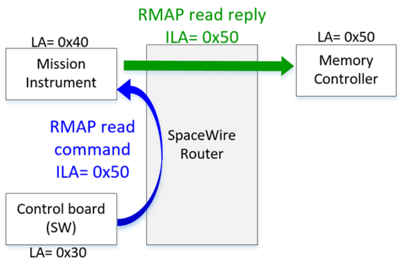


Fig.5. Example of RMAP read reply for observation data

If no RMAP reply is received from an instrument, the communication is terminated as timeout. Then, HK Telemetry is notified to ground. The RMAP command is issued to the instrument normally from next slot.

III. SPECIFIC TECHNOLOGIES

A. Protection for after Separation

When modules are separated, harnesses between the modules are cut and its cross sections exposed to space. Extra noise caused by cosmic rays and other factors can enter the harnesses and cause malfunction. Therefore, we implemented protection circuits between the parts exposed to noise, such as drivers and receivers, and the others.

SpaceWire interface subject to separation are between SMU - MDP and SMU - RDR. These two interface circuits are connected via mechanical relays as shown in Fig. 6. After separation, the exposed harness is connected to GND via a pull-down resistor by turning off the relays. This is intended to prevent malfunctions due to external noise contamination. Already, this interface has been verified at prototype test.

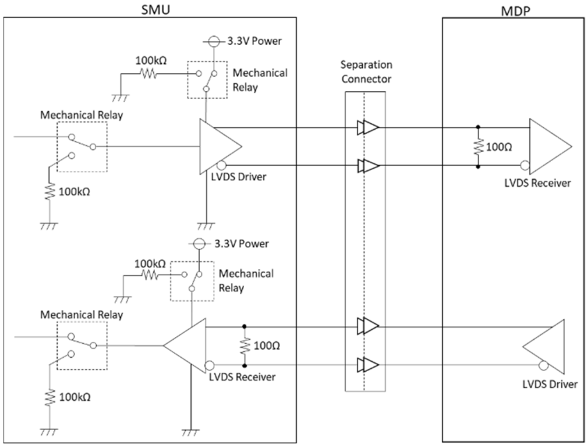


Fig. 6. Protection circuit between SMU and MDP (similar one is between SMU and RDR).

B. Strobe-less Communication

In order to return to the Earth surface, harnesses between SMU and RDR must be cut. To ensure, the number of harnesses connecting them is needed to be no more than nine. We implemented Strobe-less communication to allocate RDR power and high-speed signals to these eight harnesses. This method uses Data to realize asynchronous communication. The transmitter does not output Strobe, but only Data. The receiver latches the input Data using an internal clock (Fig. 7). The receiver will latch after detecting the data change point. If cannot be detected, the receiver will run self-propelled and acquire Data.

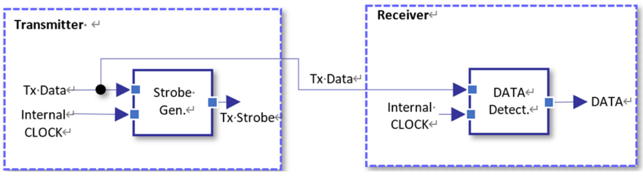


Fig. 7. H/W block of Strobe-less communication

In this concept frequency and accuracy of both of clocks must be considered. We selected oscillators with 52.5 MHz ± 50 ppm for both SMU and RDR. In our design, link rate between SMU and RDR is 10.5 Mbps, dividing this clock by 5. It is necessary to be able to detect all bits successfully when the same Data bit continues to be the most consecutive. As shown in Fig.8, it continues for a maximum of 18 bits. Therefore, it is necessary to latch the 18th bit with sufficient setup and hold time margin. This technology is useful in applications such as MMX, where the number of harnesses have severe limitations.

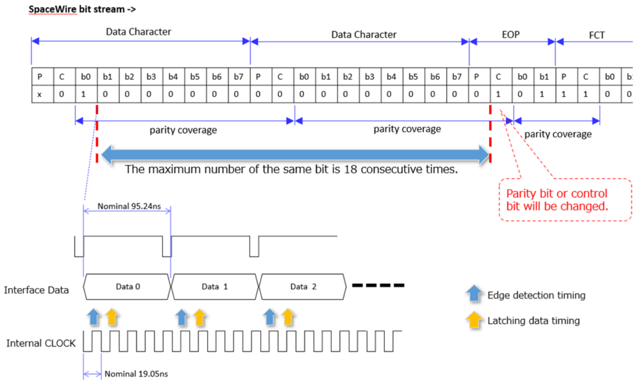

SpaceWire bit stream ->
Data Character
Data Character
EOP
FCT
P C b0 b1 b2 b3 b4 b5 b6 b7 P C b0 b1 b2 b3 b4 b5 b6 b7 P C b0 b1 P C b0 b1
x 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0
parity coverage
parity coverage
parity coverage
The maximum number of the same bit is 18 consecutive times.
Parity bit or control bit will be changed.
Nominal 95.24ns
Interface Data
Data 0
Data 1
Data 2
Edge detection timing
Latching data timing
Internal CLOCK
Nominal 19.05ns

Fig. 8. Interface timing between SMU and RDR

IV. Lessons Learned

From our development, we obtained some lessons learned.

A. Constraints of Diverting Existing Design

For SMU and MDP, our development approach was to use many existing HW and SW frameworks and to minimize new designs. As a result, the network and topology could not be simplified in some interfaces. For example, as shown in Fig.9, PTP and RMAP were mixed on the same SpaceWire line.

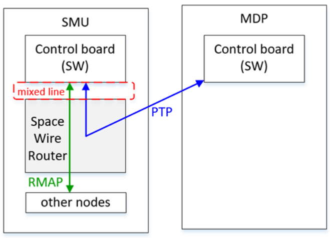


Fig. 9. Two protocols mixed SpaceWire line

In the early stages of development, it is good to determine the policy of data interface specification and feasibility of HW and SW. It is also important to choose a new design if there is a total benefit instead of relying on the existing design.

B. Interface Specification Adjustment

The total number of SpaceWire interface between mission components is eleven. Specifications such as data length, frequency of data generation, and with/without dummy data differed for each instrument. As a result, it took a lot of time to develop some component. To reduce backtracking, it is important to list up these items and determine specification values through documentation in the early stages.

For example, standardization of dummy data identification would be useful for timely development. In MMX system, how to identify whether the data is dummy or not differed from instrument to instrument. Some instruments identify it with a higher layer than CCSDS. On the other hand, some use the status field in the RMAP header.

C. Interface Verification

In order to verify the interface with each instrument, simulators of SMU and MDP were produced. We distributed them to each instrument manufacturer, and interface checks were conducted by themselves. However, we had to distribute simulators at a stage when the specifications of SMU and MDP had not yet been finalized. Moreover, because updating the simulator was not easy, verification at each manufacturer was limited. As a result, it took a lot of time for combination test with SMU/MDP and each mission equipment through the operation control system.

As an integrator, it is preferable for each manufacturer to be able to conduct verification as close to end-to-end testing as possible. If the simulator can be updated in each phase and tested by each manufacturer, including the operation control system, the combination test can be omitted or shortened.

V. Summary

First, an overview of MMX's SpaceWire network was presented. Next, we explained the techniques used to solve the unique challenges of MMX, such as circuit protection after separation and Strobe-less communication. We also reported lessons learned during the development process. MMX is currently undergoing combination tests. Through these tests, our concepts and implementation will be validated.

Acknowledgements

The author thanks the many people who assisted with the concept and offered support all along the way.

References

[1] Minoru Nakamura, Tatsuya Ito, Yasutaka Takeda, Isao Odagi, Shinya Hirakuri, Keitaro Yamagishi, Koji Shibuya, Masaharu Nomachi, " SPACEWIRE BACKPLANE WITH HIGH-SPEED SPACEFIBRE LINK ", International SpaceWire Conference 2010, St. Petersburg, Russia, June 2010

[2] Toru Sasaki, Itao Shoji, Hisayoshi Kurosawa,Tetsuro Kato, Satoshi Ichikawa, Takashi Okamoto, Taeko Seki, Mami Abe, " Application of SpaceWire to Non-Volatile Data Recorder ", International SpaceWire Conference 2013 , Gothenburg, Swede, June 2013

[3] Fujimoto et al., "JAXAs Martian Moons eXploration", European Planetary Science Congress, 2017.

[4] Kawakatsu et al., "Overview of Martian Moons eXploration (MMX)", Proceedings of the Space Science and Technology Conference 2019, JSASS-2019-4272.

Testing SpaceFibre in Orbit: the OPS-SAT and NORBY Technology Demonstrators

Marti Farras Casas
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
marti.farras@star-dundee.com

Steve Parkes
STAR-Dundee
Dundee, Scotland, UK
steve.parkes@star-dundee.com

Albert Ferrer Florit
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
albert.ferrer@star-dundee.com

Alberto Gonzalez Villafranca
STAR-Barcelona SL
Sant Cugat del Valles, Barcelona, Spain
alberto.gonzalez@star-dundee.com

***Abstract*— SpaceFibre (ECSS-E-ST-50-11C) is a technology specifically designed for use on-board spacecraft that provides point to point and networked interconnections at Gigabit rates with Quality of Service (QoS) and Fault Detection, Isolation and Recovery (FDIR). SpaceFibre is backwards compatible with SpaceWire (ECSS-E-ST-50-12C), allowing existing SpaceWire equipment to be incorporated into a SpaceFibre network without modifications at the packet level. As part of its worldwide adoption by the aerospace industry, experiments are being developed to demonstrate the capabilities and performance of SpaceFibre in space.**

This article presents the results of two collaborations of STAR-Dundee, one with the European Space Agency (ESA) OPS-SAT team and one with Thales Alenia Space (TAS) to develop SpaceFibre technology demonstrators. These consist of an implementation of the STAR-Dundee SpaceFibre Interface IP core as part of the spacecraft payload. The aim of these collaborations was to increase the technology readiness level (TRL) of SpaceFibre by demonstrating an operational SpaceFibre link in orbit, providing examples of flying heritage for the technology.

The first collaboration is an implementation of a SpaceFibre link in a commercial off-the-shelf (COTS) device hosted in the OPS-SAT spacecraft developed by ESA. The entire SpaceFibre was implemented in the FPGA (Intel Cyclone V) and it was controlled and monitored from a CPU. The experiment generates and sends data in loopback using different virtual channels in the link, and the received data is subsequently checked for errors. During the activity SpaceFibre is continuously being monitored looking for issues in the link.

The second collaboration was with TAS in the NORBY mission. In this activity SpaceFibre is also implemented in a commercial FPGA, and the monitoring and control of the link is done by a LEON3 processor. Similarly, this activity also uses data generators to send data in loopback and the received data is checked for errors.

The results of both activities were successful. All the data transmitted were received with no errors, showing that SpaceFibre links implemented even in commercial FPGAs can reliably operate in space.

Keywords—SpaceFibre, OPS-SAT, NORBY

I. Introduction

SpaceFibre (SpFi) is a very high-speed serial link designed specifically for use onboard spacecraft [1]. It was released as an ECSS standard in 2019. SpFi can operate over fibre-optic and electrical cable, and aims to complement the capabilities of the widely used SpaceWire (SpW) onboard networking standard [2]. It improves the data rate by a factor of more than 10, reducing the cable mass and providing galvanic isolation. SpFi provides a coherent QoS mechanism able to support bandwidth reserved, scheduled and priority-based qualities of service. It also substantially improves the FDIR capabilities compared to SpW.

SpFi is being adopted worldwide by the aerospace industry and as part of this adoption, new experiments are developed to demonstrate its capabilities and performance in real space environments. This paper describes two experiments designed for two different missions that implemented and run a demonstrator design of the SpFi interface in orbit.

A. OPS-SAT Mission

OPS-SAT is a 3U CubeSat (Fig. 1) launched by ESA on December 18, 2019 [3]. It is the first nanosatellite to be directly owned and operated by ESA. OPS-SAT includes a system on module (SoM) platform containing state-of-the-art semiconductor technologies. The platform is called satellite experimenter processing platform (SEPP) and can take control over of the whole satellite and doing intense processing in parallel. The SEPP features a 925 MHz dual-core ARM Cortex A9 Hard Processor System (HPS), an integrated Altera Cyclone V COTS FPGA and 16 GB of flash storage.

B. NORBY Mission

NORBY is a 6U CubeSat nanosatellite (Fig. 2) by Novosibirsk State University launched on September 28, 2020 [4]. The NORBY CubeSat platform is designed to create relatively inexpensive specialised nanosatellites with a payload targeted for scientific, technological, and commercial applications. The main goals of the NORBY launch are flight tests of the platform, verification of its functional capabilities and technical solutions under real conditions in low-Earth orbit, as well as carrying out scientific and technological research envisaged by the nanosatellite payload program. The NORBY platform connects a commercial FPGA with a LEON3 processor. The SpFi experiment was manufactured by an international cooperation between Information Satellite Systems – Reshetnev Company and Thales Alenia Space Spain.

II. SpaceFibre Technology Demonstrator

This section describes the system used for the SpFi demonstrator. Some parts are shared between both missions, and some are designed according to the specifics of each mission. The main goal of the experiments was to transmit and check as much data as possible in the timeslot in which the experiment was allowed to run.

A. SpaceFibre Demonstrator Overview

The following subsections provide an overview of the elements in the demonstrator shared between the missions. Figure 3 depicts the block diagram of the SpFi demonstrator used in both missions.

1) Software Application

While a software application is specifically designed for each mission, the goal of the software application is the same: to control and monitor the SpFi link and store the relevant results to be sent down as telemetry. The application ensures that, at the start of the experiment, the SpFi link is correctly started and configured.

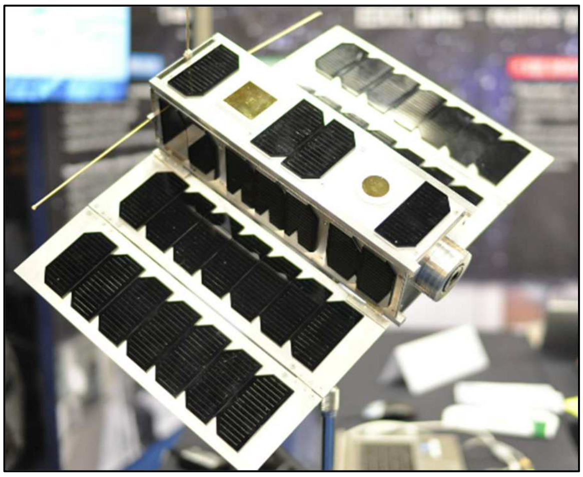


Fig. 1. The OPS-SAT satellite.

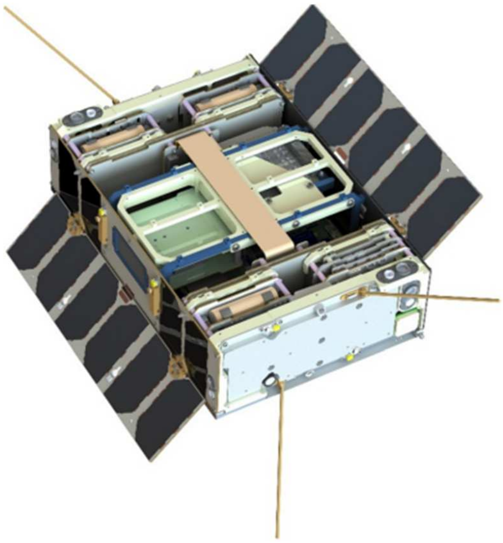


Fig. 2. The NORBY satellite.

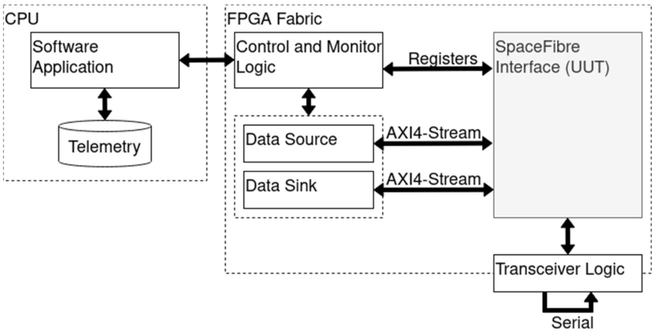


Fig. 3. SpFi demonstrator block diagram.

The low complexity of the software application is a consequence of a key feature of the SpFi protocol, which is its ease of use. SpFi just requires enabling the link and after few microseconds (typically ~40 µsec), data can be transferred at full data rate.

2) Control and Monitor Logic

This block acts as the interface between the software application and the hardware design. The information from/to the software application is stored in registers which are connected to the status and control ports of the SpFi interface, and data source/sink components.

3) Data Source and Sink

All virtual channels (VC) are connected to independent data source and data sink modules. Each transmit VC can receive data from a Source capable of providing data at the maximum rate supported by the SpFi link. This way a single VC can saturate the SpFi link with data if required. The data source generates SpFi packets. The data pattern consists of a 16-bit counter increased by 1 every transmitted data word. As the word is 32-bit wide, the value of the counter is duplicated. The counter is initialised to 0 (i.e. 0x0000) and after reaching its maximum value (i.e. 0xFFFF) it starts back from 0 again.

When the data sink detects an incorrect data pattern it reports a data error. When the sink detects an error end-of-packet (EEP), this condition is also reported. In the occurrence of an EEP or data error the data sink automatically resynchronises with the next valid word. This avoids reporting continuous false error detections whenever a packet has been truncated (EEP) or lost due to the source and the sink being out of sync.

4) Transceiver Logic

This block contains the logic to control the transceiver of the FPGA. This logic is specific to each mission to target the appropriate technology. It includes the configuration and connection of the specific clocks used by each system. Due to the SpFi experiments being adopted at a late stage on both missions, a fully representative implementation of the SpFi links was not possible. Instead, a loopback connection was used.

5) SpaceFibre Interface

This is the unit under test (UUT). It instantiates the STAR-Dundee SpFi Single-Lane Interface IP core using the configuration parameters required for each technology. For both missions the SpFi link uses two VCs to transmit and receive data at lane rate of 2.5 Gbit/s. Each SpFi VC uses an

AXI4-Stream interface to connect to the data source and sink.

B. OPS-SAT Experiment

When OPS-SAT was in its definition phase, it was not envisaged that the Altera FPGA transceivers would be required. Therefore, the transceivers pins of the FPGA were not connected to the PCB. This prevented the SpFi experiment from using the transceiver external reference clock pins, and a custom clock scheme was put in place to fix this issue.

Listed below are the main settings used in the demonstrator system for OPS-SAT:

- SpFi interface configured to use a 20-bit parallel transceiver interface.
- The interface used between the CPU and the FPGA is the Avalon Memory Mapped Interface.
- The loopback is done at the physical medium attachment (PMA).
- Included logic to also transmit, receive and check SpFi broadcast messages. The experiment transmits a broadcast message every 1 µsec.
- The software application applied a reset of the experiment every minute. The reason was to exercise more logic of the interface during the experiment and, in case of a persistent error, recover the link and continue the experiment.

C. NORBY Experiment

The NORBY mission imposed a strict constraint in the amount of telemetry space available for downloading the results of the experiment. A summary of the main parameters of the demonstrator was thus generated for download.

Listed below are the main settings used in the demonstrator for NORBY:

- SpFi interface configured to use a 32-bit transceiver interface.
- The interface used between the CPU and the FPGA is an APB interface.
- The initial approach was to perform a loopback outside the transceiver, i.e. serial loopback. Due to design limitations, the loopback was done inside the SpFi interface using its inbuilt loopback feature. This is a parallel loopback and not a serial loopback.
- The software application restarted the experiment at the beginning of each run. The duration of each run was 16 seconds.

III. Results

Both missions reached orbit successfully, and the SpFi experiments were correctly executed. This section presents the results of the experiments extracted from the telemetry received from each satellite.

A. OPS-SAT

The experiment was run in three separate campaigns with a duration of 10, 40 and 40 minutes, for a total runtime time of 90 minutes. The resulting telemetry was parsed to ensure that during each run, after the SpFi link was ready, the data generators and checkers were transmitting and receiving data at the maxim available data rate, each VC sharing almost 50% of the bandwidth. About 1% of the link bandwidth was reserved for the broadcast messages. In parallel, the link was monitored to detect anomalies in its operation, such as transient errors, but in the entire runtime no error was detected in the link.

B. NORBY

The experiment had a fixed runtime of 15.5 seconds per run. The experiment runs were launched in sets of three for a total of 92 runs. The total runtime of the experiment was around 24 minutes.

The resulting telemetry was parsed to ensure that during each run, after the SpFi link was ready, the data generators and checkers were transmitting and receiving data as expected. Due to the limits on the size of the telemetry the bandwidth utilization was extracted from the status of the VC buffers, by checking that both transmit and receive buffers reported data filling them during the experiment. In parallel, the link was monitored to detect any anomaly that could appear, such as transient errors, but in the entire runtime no error was detected in the link.

IV. Conclusion

The first publicly known missions to have tested SpaceFibre in orbit have been described. The STAR-Dundee SpaceFibre Interface IP has flown onboard OPS-SAT and NORBY. The successful results of both experiments demonstrate SpaceFibre operating in orbit. The STAR-Dundee SpaceFibre IP family [5] has been, and is currently being, implemented in FPGA and ASIC designs for several missions and products in Europe and the USA.

Acknowledgment

STAR-Dundee would like to thank the teams of both OPS-SAT (ESA) and NORBY (TAS-Spain and ISS) missions for his support during the development and execution of these test campaigns.

References

[1] ECSS Standard ECSS-E-ST-50-11C, “SpaceFibre – Very high-speed serial link”, European Cooperation for Space Data Standardization, 15th May 2020. Available from http://www.ecss.nl.

[2] ECSS Standard ECSS-E-ST-50-12C, “SpaceWire, Links, Nodes, Routers and Networks”, Issue 1, European Cooperation for Space Data Standardization, July 2008, available from http://www.ecss.nl.

[3] OPS-SAT information page. Available online at: https://www.esa.int/Enabling_Support/Operations/OPS-SAT.

[4] V. Yu Prokopyev et al., “NORBY CubeSat nanosatellite: design challenges and the first flight data”, 2021 J. Phys.: Conf. Ser. 1867.

A. Gonzalez Villafranca, A. Ferrer Florit, M. Farras Casas and S. Parkes, “SpaceFibre IP Cores for the Next Generation of Radiation-Tolerant FPGAs”, 2022 International SpaceWire Conference.

Papers Indexed by Author

Author Surname A - M

Author Surname M – Z

Papers Indexed by Session

Monday 17th October

Networks & Protocols (Short Papers)

Tuesday 18th October

Networks & Protocols 1 (Long Papers)

Test & Verification (Long Papers)

Onboard Equipment (Long Papers)

Poster Presentations

Components (Long Papers)

Wednesday 19th October

Components (Short Papers)

Test & Verification (Short Papers)

Networks & Protocols 2 (Long Papers)

Missions & Applications (Short Papers)

Exhibitors

4LINKS LTD.

4Links designs and manufactures a comprehensive range of SpaceWire test equipment. These provide unequalled insight into the operation of your SpaceWire network. Whether you are monitoring SpaceWire traffic between two devices, or creating a device model to initiate or respond to another device, 4Links offers you the right solution, so you can quickly and efficiently find and diagnose your problem. At the SpaceWire and SpaceFibre Conference 4Links will introduce and demonstrate its new Loki product. The concept is based around the same boards being usable across a range of user defined solutions. The Loki FPGA Mezzanine board uses an FPGA with configuration and local memory devices built into it. The functionality of the FPGA is determined by the FPGA image loaded by the configuration memory. The Loki IO Carrier board uses the same physical IO Carrier PCB, but with different build options depending on the required function. It can be configured as a CCSDS/RS422 board, a SpaceWire Ethernet Bridge (SEB), or a SpaceWire Router Switch (SRS) boards. It connects to the same FPGA Mezzanine, but with different FPGA images. 4Links is based in Shenley Wood, Milton Keynes, Buckinghamshire, UK.

AXON’ CABLE

Involved in numerous space projects in orbit or beyond for over 20 years, Axon' Cable has a large experience in designing and manufacturing interconnect solutions able to withstand the stresses of launch and the harsh space environment. Lightweight, miniature, reliable and highly resistant to cosmic radiation and high temperatures: these are the qualities of wires, cables, cable assemblies, wire harnesses and connectors offered by Axon’ for space applications.

From material choice or components, to the design, routing, manufacturing, test and product qualification, Axon’ offers a complete service for the cabling of spacecraft or rover vehicles. 2D or 3D cabling boards can be developed to ensure perfect integration at the customer’s facilities.

The company is particularly strong in the area of high data rate interconnect, and has provided high speed links on-board telecommunication, military, scientific and earth observation satellites including Alphabus, ISS, Astro H, Bepi Columbo, Maven, Sentinel and Solar Orbiter.

CAES
PIONEERING ADVANCED ELECTRONICS

COBHAM GAISLER

We are a world leader in embedded computer systems for harsh environments, with footprints in many parts of the solar system.
We provide the complete ecosystem to support digital hardware design for mission critical System-on-Chip solutions. The IP cores and development tools support processors based on the SPARC and RISC-V architectures, also complemented by a software ecosystem that includes debugging tools, simulators, compilers, operative systems and bootloaders. In addition, we have a long experience in the management of ASIC development projects and the design of flight quality microelectronic devices: we provide several radiation-hardened standard components.

Glenair®

GLENAIR

Glenair offers a full-spectrum product line designed to meet every interconnect requirement including a broad range of harsh environment connectors, cable assemblies, wiring harnesses, conduit, braid and related accessories. Our products are used in diverse markets including space, avionics, defence and more. Glenair's photonic and fibre optic solutions include optoelectronic connector contacts and Space Fibre compliant digital transceivers addressing data rates from a few Mb/s to ribbon optical fibre-based solutions to 100Gb/s. Our space grade high power (up to 10W) optical amplifiers combined with our high-power handling optical connectors and DWDM optical transceivers offer robust solutions for free space optical inter-satellite links for applications ranging from CubeSats to high-throughput telecom satellites. Extensive radiation testing has been conducted on Glenair transceivers including proton, heavy-ion, gamma and neutron to extreme levels (reports available). Additionally, we offer a design, build and test service for complex high-speed fibre-optic and electrical space harnesses for onboard satellite interconnections and for ground testing of satellite systems. These can incorporate nano-D, micro-D and our new GMMD and RF connector systems. Finally, Glenair hold down release mechanisms (HRDMs) offer pyrotechnic-free and user serviceable solutions for on-orbit deployment of space payloads.

INGENIARS

IngeniArs was founded in 2014 as an innovative start-up and University of Pisa spin-off company, from the long experience of our co-founders in the area of Electronics and Computer Science. We are specialised in the design and development of innovative high-tech electronic/informatics systems in the domains of Aerospace, Artificial Intelligence, Healthcare, and Cybersecurity. We are able to manage the full lifecycle of electronics, microelectronics, embedded systems, smart sensors, web applications and services, and we offer high-quality products and services to our customers and partners. We are Microchip Design Partner, Xilinx Certified Partner, Associate Member of the CCSDS and we are part of the NVIDIA Inception Partnership Program. Our company is UNI EN ISO 9001:2015 and UNI EN ISO 13485:2016 certified

ROHDE & SCHWARZ

With its extensive product portfolio, the company makes an important contribution to a safer and connected world. In the test & measurement, secure communications, networks & cybersecurity and broadcast & media markets, customers worldwide rely on Rohde & Schwarz and its cutting-edge solutions. In addition to its established business fields, Rohde & Schwarz has made substantial investments in future technologies such as artificial intelligence, the industrial internet of things (IIoT), 6G, cloud solutions and quantum technology. Founded more than 85 years ago, the group is a reliable partner for industry and government customers around the globe, with more than 13,000 employees worldwide working in more than 70 countries.

SMITHS INTERCONNECT

Smiths Interconnect is a leading provider of technically differentiated electronic components, subsystems, microwave, optical and radio frequency products that connect, protect and control critical applications in the commercial aviation, defense, space, communications and industrial market segments. Smiths Interconnect is synonymous with exceptional performance whenever a technologically advanced, high quality solution is required to ensure reliability and safety.
Smiths Interconnect is an approved vendor for international space agencies including ESA, ISRO, JAXA and NASA, and has proudly delivered failure-free performance in numerous spaceflight programs. We work globally with our customers and space agencies to design the next generation of solutions for launchers, satellites, manned space flight and ground systems support.

STAR-DUNDEE LTD.

STAR-Dundee is a leading supplier of spacecraft on-board data-handling technology with significant SpaceWire and SpaceFibre experience and expertise, this year celebrating our 20th anniversary. We supply a comprehensive range of SpaceWire and SpaceFibre IP cores and test and development equipment to the international aerospace industry.

Our highly experienced engineers were instrumental in the development of SpaceWire, writing the standard with input from international engineers. Our SpaceWire IP is now widely used and integrated in spaceflight systems monitoring the Earth, exploring our Solar System, studying the universe and supporting commercial space applications.

Our engineers led the research, development and standardisation of SpaceFibre, the next generation of SpaceWire. SpaceFibre offers higher data rates, quality of service, FDIR, deterministic delivery, low latency time-synchronisation and event signalling, while being backwards compatible with SpaceWire at the packet level. Our SpaceFibre IP cores have flown on in-orbit demonstration missions and are being implemented in spaceflight systems

SYSTEM-ON-CHIP ENGINEERING S.L

SoC-e is a worldwide leading supplier of Ethernet and SpaceWire communication solutions based on FPGA technology.

SoC-e is a pioneer in developing a portfolio of IP cores that implement the leading-edge networking, synchronization , and security technologies for critical systems.

This SoC-e technology has been applied in more than 100 projects worldwide in very different applications for the Space, Industrial, Energy, Medical and Aerospace sectors. Multinationals and SME companies integrate SoC-e solutions for Space (SpaceWire), Networking (Ethernet Switch, TSN), and wire-speed cryptography implementations to secure real-time traffic.

ASI Agenzia Spaziale Italiana

esa

INGENIArs

STAR-Dundee

IEEE

AESS

Università di Pisa