Ultra-Fast, Low-Overhead SpaceFibre Communication for System-on-Chips

Dave Gibson
STAR-Dundee Ltd.
Dundee, Scotland, UK
david.gibson@star-dundee.com

Andrew MacLennan
STAR-Dundee Ltd.
Dundee, Scotland, UK
andrew.maclennan@star-dundee.com

Stuart Mills
STAR-Dundee Ltd.
Dundee, Scotland, UK
stuart.mills@star-dundee.com

Steve Parkes
STAR-Dundee Ltd.
Dundee, Scotland, UK
steve.parkes@star-dundee.com

Albert Ferrer Florit
STAR-Barcelona S.L.
Barcelona, Spain
albert.ferrer@star-dundee.com

Alberto Gonzalez Villafranca
STAR-Barcelona S.L.
Barcelona, Spain
alberto.gonzalez@star-dundee.com

***Abstract*— A SpaceFibre [1] [2] Processor Endpoint has been developed to provide ultra-fast SpaceFibre communication for System-on-Chips (SoCs) with processors running an operating system such as Linux. The SpaceFibre Processor Endpoint provides a hardware-accelerated interface between user-space software applications and a SpaceFibre network. This approach significantly offloads work from the processor, enabling data rates in and out of user-space memory at tens of Gbit/s with minimal processor overhead.**

SpaceFibre is the next-generation of SpaceWire [3] [4], providing multi-Gbit/s on-board networking; novel quality-of-service including priority, bandwidth reservation and scheduling; built-in fault detection, isolation and recovery; and low-latency broadcast messaging for time-distribution, synchronisation, event signalling and error notification. At the packet level, SpaceFibre is backwards compatible with SpaceWire, making it straightforward to integrate existing SpaceWire equipment into a SpaceFibre network.

SpaceFibre supports very high data rates using multi-lane links. For example, in recent work [5], an experimental SpaceFibre link was demonstrated at 100 Gbit/s in an AMD Versal Adaptive SoC [6] under heavy-ion radiation testing using a quad-lane link with each lane operating at 25 Gbit/s.

The SpaceFibre Processor Endpoint has been developed to support these very high data rates efficiently. This paper first introduces the SpaceFibre Processor Endpoint and software. Next, it describes the use of the SpaceFibre Processor Endpoint in different platforms. Finally, performance results are provided, illustrating the benefits of using the SpaceFibre Processor Endpoint.

Keywords*— *SpaceFibre, On-board Networking, On-board Data Handling, High-Performance, Hardware-Acceleration, Software, System-on-Chips, Embedded Systems

I. Introduction

The primary objective of the SpaceFibre Processor Endpoint is to provide a rapid and simple way to integrate one or more SpaceFibre interfaces into an on-board processing system, with full software support designed to provide maximum performance with minimal processor overhead.

In previous work [7], Remote Direct Memory Access (RDMA) over SpaceFibre was implemented for Linux using an AMD Zynq UltraScale+ MPSoC (ZCU102) development board [8]. The SpaceFibre link in that implementation was a quad-lane link running at a lane rate of 7.8125 Gbit/s, providing a link rate of 31.25 Gbit/s, and a maximum user data rate of 23.8 Gbit/s. Performance results showed approximately 23 Gbit/s data rates while using less than 1% processor utilisation.

The SpaceFibre Processor Endpoint follows on from the RDMA over SpaceFibre activity, using similar hardware-accelerated techniques. However, it does not require a specific packet format, which allows it to be used to communicate with any SpaceFibre system, not just those that implement RDMA over SpaceFibre. The SpaceFibre Processor Endpoint is also targeting much higher link rates and includes additional capabilities including low-latency software support for sending and receiving SpaceFibre broadcast messages, as well as support for SpaceWire interfaces. The SpaceFibre broadcast message capability can be used for various purposes including, for example, receiving status information or error notifications from other devices, or sending trigger commands to other devices.

II. Target Platforms

The initial platforms that the SpaceFibre Processor Endpoint has been implemented in include AMD's Zynq UltraScale+ MPSoC and Microchip's PolarFire SoC [9], and it will be tested in AMD's Versal Adaptive SoC in the near future.

For these platforms, the SpaceFibre Processor Endpoint IP core is implemented in the Programmable Logic (PL) of the SoC, with the software running in Linux on the Processing System (PS). A device-tree entry is then used to make the SpaceFibre Processor Endpoint available to the supporting Linux platform driver.

Alternatively, the SpaceFibre Processor Endpoint can also be connected to other processor systems by implementation in a standalone Field Programmable Gate Array (FPGA) such as Microchip's PolarFire FPGA, with the software running in Linux on an external processor. A high-speed interconnect in the FPGA such as Peripheral Component Interconnect Express (PCIe) is then used to make the SpaceFibre Processor Endpoint available to the supporting Linux PCIe driver.

In addition to the Linux version, a bare-metal version of the software is being developed to allow the SpaceFibre Processor Endpoint to be used in applications where a full operating system is not required or possible.

A photograph showing the SpaceFibre Processor Endpoint implemented in a Microchip PolarFire FPGA Evaluation Kit (MPF300) [10] is provided in Fig. 1.

Fig. 1. SpaceFibre Processor Endpoint on Microchip PolarFire

In Fig. 1, the SpaceFibre Processor Endpoint is implemented in the Microchip PolarFire FPGA, connected to an external processor over PCIe via an extender cable, and connected to the SpaceFibre network using a Small Form-Factor Pluggable Plus (SFP+) cable assembly.

Another photograph showing the SpaceFibre Processor Endpoint implemented in an AMD Zynq UltraScale+ MPSoC (ZCU102) development board is provided in Fig. 2.

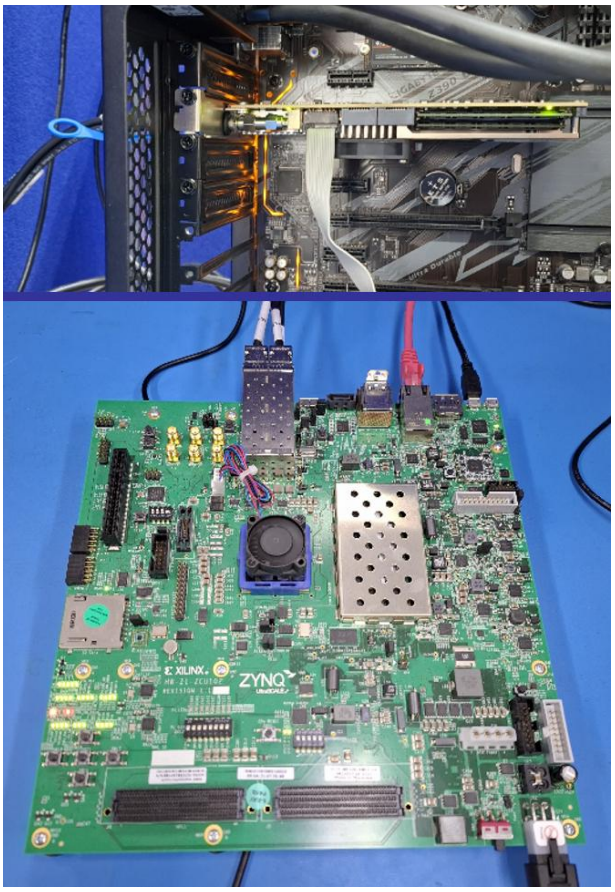

Fig. 2. SpaceFibre Processor Endpoint on AMD Zynq UltraScale+ MPSoC (ZCU102)

In Fig. 2, the SpaceFibre Processor Endpoint is implemented in the PL of the AMD Zynq UltraScale+ MPSoC (ZCU102), with the software running in PetaLinux 2022.1 in the ARM Cortex-A53 Central Processing Unit (CPU) of the PS. In this case, the SpaceFibre Processor Endpoint has a quad-lane SpaceFibre link, with a lane signalling rate of 7.5 Gbit/s, resulting in a link rate of 30 Gbit/s. The SpaceFibre Processor Endpoint is connected to a STAR-Ultra PCIe Interface [11] board using a Quad SFP+ (QSFP+) to 4xSFP+ cable assembly.

III. Architecture

A high-level block diagram illustrating the architecture of the SpaceFibre Processor Endpoint is provided in Fig. 3.

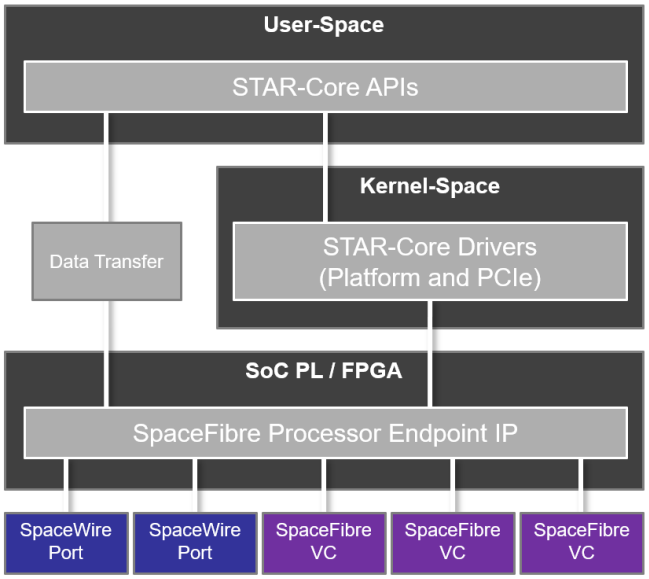

Fig. 3. SpaceFibre Processor Endpoint Architecture

In Fig. 3, the architecture is divided into four layers, from the network to the user applications.

In the bottom layer, the SpaceFibre Processor Endpoint consists of a configurable number of SpaceWire ports and a configurable number of SpaceFibre Virtual Channels (VCs), which are contained in one or more SpaceFibre ports.

In the next layer, the SpaceFibre Processor Endpoint IP provides interfaces to the SpaceWire ports and/or SpaceFibre ports and VCs. These interfaces are used to transfer packets, time-codes (for SpaceWire ports) and broadcast messages (for SpaceFibre ports) to and from the SpaceFibre Processor Endpoint IP and the SpaceWire and/or SpaceFibre networks.

Above the SpaceFibre Processor Endpoint IP is the kernel-space drivers, which provide interfaces between the host processor and the SpaceFibre Processor Endpoint IP. There is a small hardware abstraction layer in the core of the kernel-space drivers that handles the differences between devices connected by a PCIe interface (for standalone FPGAs) and devices connected by a platform interface (for SoCs).

Finally, above the kernel-space drivers is the user-space Application Programming Interfaces (APIs), which provide an interface between the user applications and the kernel-space drivers. Additionally, the kernel-space drivers set up the direct path between the user-space APIs and the SpaceFibre Processor Endpoint IP.

IV. Performance Testing

Performance testing of the SpaceFibre Processor Endpoint was conducted to measure the data rates and CPU utilisation when sending and receiving SpaceFibre packets of increasing lengths and varying strategies.

The following sections describe the performance test setup, procedures, and results.

A. Performance Test Setup

A high-level block diagram illustrating the performance test setup is provided in Fig. 4.

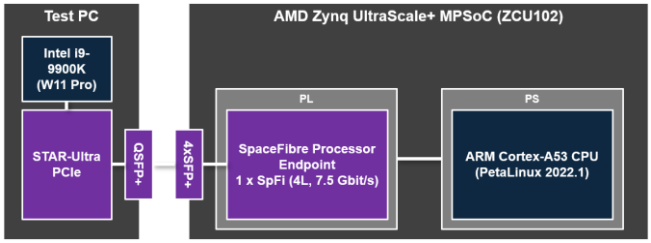

Fig. 4. Performance Test Setup

In Fig. 4, the block on the left is the Test PC which has the following specification:

- Intel i9-9900K CPU
- 128 GB DDR4 memory
- 512 GB NVMe storage
- Windows 11 Pro

The Test PC contains a STAR-Ultra PCIe, which is a SpaceFibre test and development product with two quad-lane SpaceFibre interfaces and a PCIe Gen 3 eight-lane interface to the host. The STAR-Ultra PCIe uses the STAR-System PCIe driver and APIs [12] to send and receive SpaceFibre packets.

The block on the right of Fig. 4 is the AMD Zynq UltraScale+ MPSoC (ZCU102), which has an ARM Cortex-A53 CPU in the PS, and the SpaceFibre Processor Endpoint implemented in the PL, providing a quad-lane SpaceFibre interface. The ZCU102 uses the SpaceFibre Processor Endpoint platform driver and APIs to send and receive SpaceFibre packets.

The quad-lane SpaceFibre link between the STAR-Ultra PCIe and the SpaceFibre Processor Endpoint is provided using a QSFP+ to 4xSFP+ cable assembly between the QSFP+ connector on the STAR-Ultra PCIe, and the SFP+ connectors on the ZCU102.

The lane signalling rate is configured to 7.5 Gbit/s on the STAR-Ultra PCIe and the SpaceFibre Processor Endpoint, providing an overall link signalling rate of 30 Gbit/s. With a 30 Gbit/s link signalling rate, the maximum unidirectional data rate is approximately 22.9 Gbit/s, and the maximum bidirectional data rate is approximately 22.2 Gbit/s.

B. Performance Test Procedures

Performance tests were conducted using the following test scenarios:

- Sending single packets at a time from the SpaceFibre Processor Endpoint, received by the STAR-Ultra PCIe.
- Sending multiple packets at a time from the SpaceFibre Processor Endpoint, received by the STAR-Ultra PCIe.
- Receiving single packets at a time on the SpaceFibre Processor Endpoint, sent by the STAR-Ultra PCIe.
- Receiving multiple packets at a time on the SpaceFibre Processor Endpoint, sent by the STAR-Ultra PCIe.

For all scenarios, the performance tests were conducted for a range of packet lengths, from 1 KB to 256 KB, increasing in powers of 2. At each packet size, the scenarios were repeated for 10 iterations of 10 seconds each.

During the execution of the performance tests, the data rate was measured using the total number of bytes sent or received on the SpaceFibre Processor Endpoint, and CPU utilisation was measured using the Linux kernel/system statistics files.

C. Performance Test Results

The following sections provide results for the performance test scenarios listed in the previous section.

1) Sending Single Packets

When sending single packets with the SpaceFibre Processor Endpoint, there is no per-byte copy overhead for the packet data, as data is read by the SpaceFibre Processor Endpoint directly from user-space memory. The overhead consists of a per-packet cost to submit the send operations and handle the interrupts signalling the completion of the send operations.

For this test scenario, each packet is submitted and waited on for its completion before moving on to the next packet.

The performance results for sending single packets at a time are illustrated in Fig. 5.

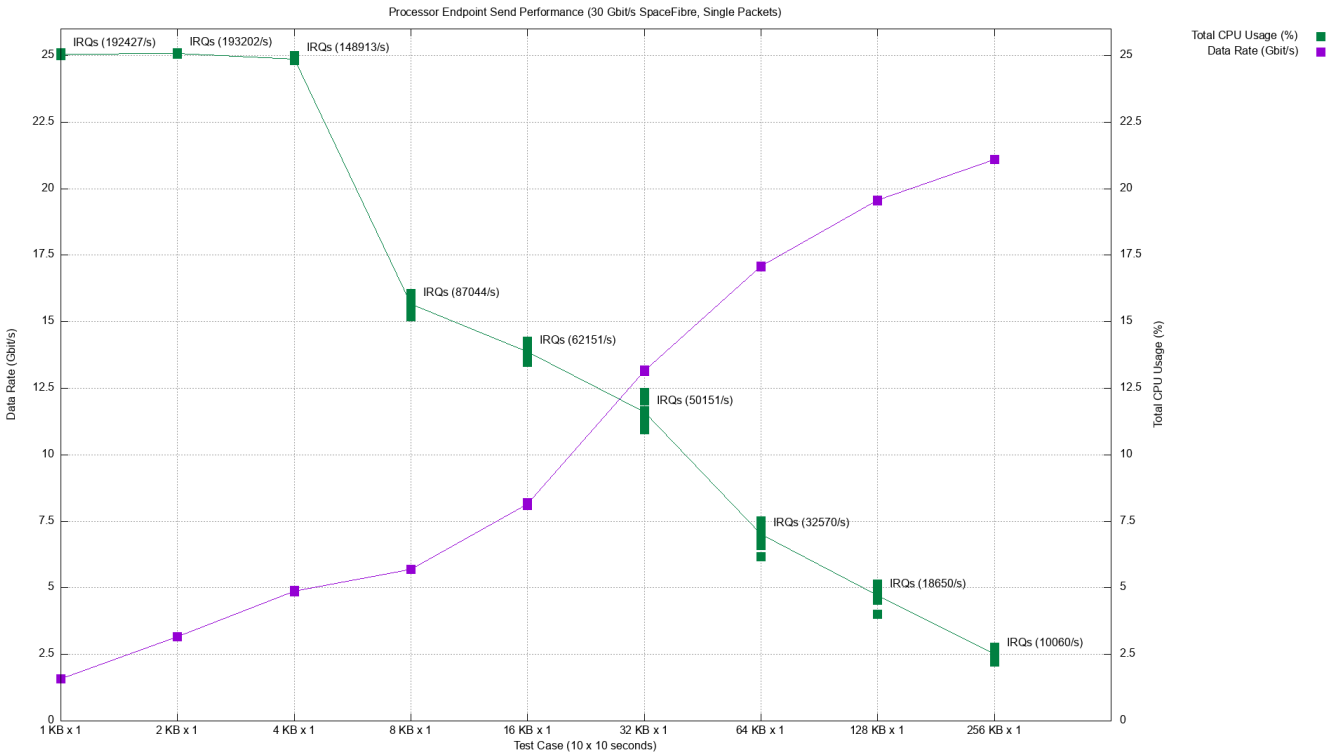

Fig. 5. Sending Single Packets Chart

In Fig. 5 (and subsequent performance charts), the data rate (Gbit/s) is plotted in purple, and the CPU utilisation (%) is plotted in green. The left-vertical axis shows the data rate, and the right-vertical axis shows the CPU utilisation. The horizonal axis shows the packet length (KB), increasing from left to right. For each test case, the results for each iteration are drawn as points, and the lines are drawn through the average of those points.

As shown in Fig. 5, when sending single packets, the CPU utilisation is initially high because, as described above, there is an overhead to submit each send operation and process its completion interrupt. As the packet length increases, the CPU utilisation reduces, and the data rate increases, because there are fewer, larger packets being sent every second.

When the packet length reaches 256 KB, the data rate achieved is over 20 Gbit/s, and the CPU utilisation is approximately 2.5%. The intention of this scenario is to demonstrate that high performance can be achieved even with the simplest approach.

The performance results for sending single packets at a time are listed in Table I, with the average values across all iterations, followed by the worst-case in parentheses.

TABLE I. SENDING SINGLE PACKETS TABLE

Test Case	Data Rate (Gbit/s)	CPU Utilisation (%)
1 KB	1.57 (1.57)	25.06 (25.10)
2 KB	3.16 (3.16)	25.09 (25.10)
4 KB	4.87 (4.86)	24.86 (24.99)
8 KB	5.70 (5.70)	15.68 (16.06)
16 KB	8.13 (8.08)	13.80 (14.27)
32 KB	13.14 (13.12)	11.48 (12.35)
64 KB	17.07 (17.05)	7.09 (7.51)
128 KB	19.55 (19.54)	4.70 (5.15)
256 KB	21.09 (21.09)	2.49 (2.78)

2) Sending Multiple Packets

To increase the data rate and reduce the CPU utilisation when sending packets, the per-packet overhead needs to be reduced. This can be done simply by grouping multiple packets together into single operations. Doing this coalesces the completion interrupts together and changes the per-packet overhead to a per-group overhead, meaning there is one send operation submission for a group of packets, and one interrupt signalling their completion.

For this test scenario, the group size is set to either 128 or the maximum number of packets that fit within a 1 MB buffer e.g., 8 KB x 128, 16 KB x 64, and so on. Each group of packets is submitted and waited on for its completion before moving on to the next group.

The performance results for sending multiple packets at a time are illustrated in Fig. 6.

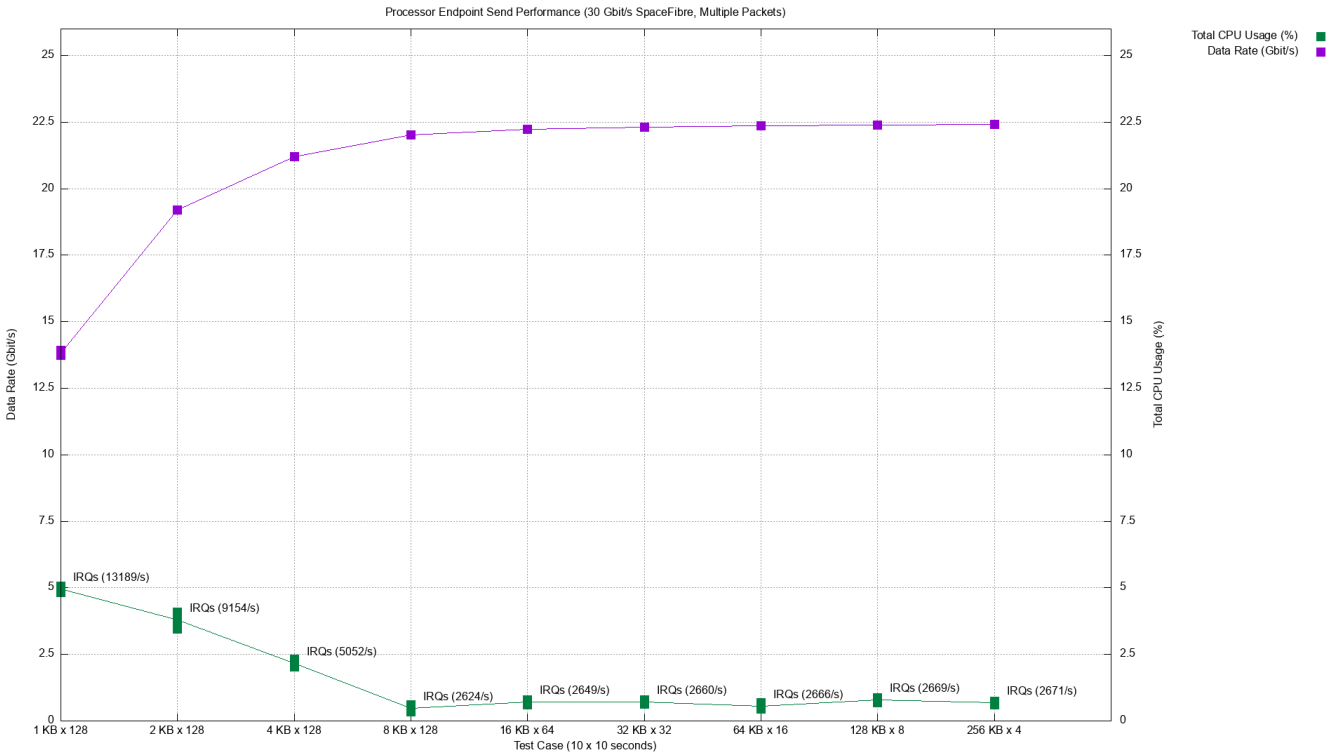

Fig. 6. Sending Multiple Packets Chart

As shown in Fig. 6, the data rate and CPU utilisation results starting at 1 KB packets are significantly better than the results in Fig. 5, because the packets are being sent in groups of 128 at a time. This trend continues until the data rate saturates at just over 22 Gbit/s with a CPU utilisation of less than 1%.

The performance results for sending multiple packets at a time are listed in Table II, with the average values across all iterations, followed by the worst-case in parentheses.

TABLE II. SENDING SINGLE PACKETS TABLE

Test Case	Data Rate (Gbit/s)	CPU Utilisation (%)
1 KB x 128	13.83 (13.73)	4.94 (5.06)
2 KB x 128	19.19 (19.19)	3.79 (4.10)
4 KB x 128	21.19 (21.19)	2.15 (2.33)
8 KB x 128	22.01 (22.01)	0.51 (0.61)
16 KB x 64	22.22 (22.21)	0.71 (0.81)
32 KB x 32	22.31 (22.31)	0.73 (0.80)
64 KB x 16	22.36 (22.36)	0.56 (0.68)
128 KB x 8	22.39 (22.39)	0.79 (0.88)
256 KB x 4	22.41 (22.40)	0.68 (0.75)

3) Receiving Single Packets

When receiving packets with the SpaceFibre Processor Endpoint, similar to sending packets, there is no per-byte copy overhead for the packet data, as data is written by the SpaceFibre Processor Endpoint directly to user-space memory. The overhead depends on the incoming traffic pattern, as the receiving software will either consume immediately available data, or it will block waiting for an interrupt signalling that there is new data available.

For this test scenario, the completion for each packet is processed one at a time before moving on to the next packet.

The performance results for receiving single packets at a time are illustrated in Fig. 7.

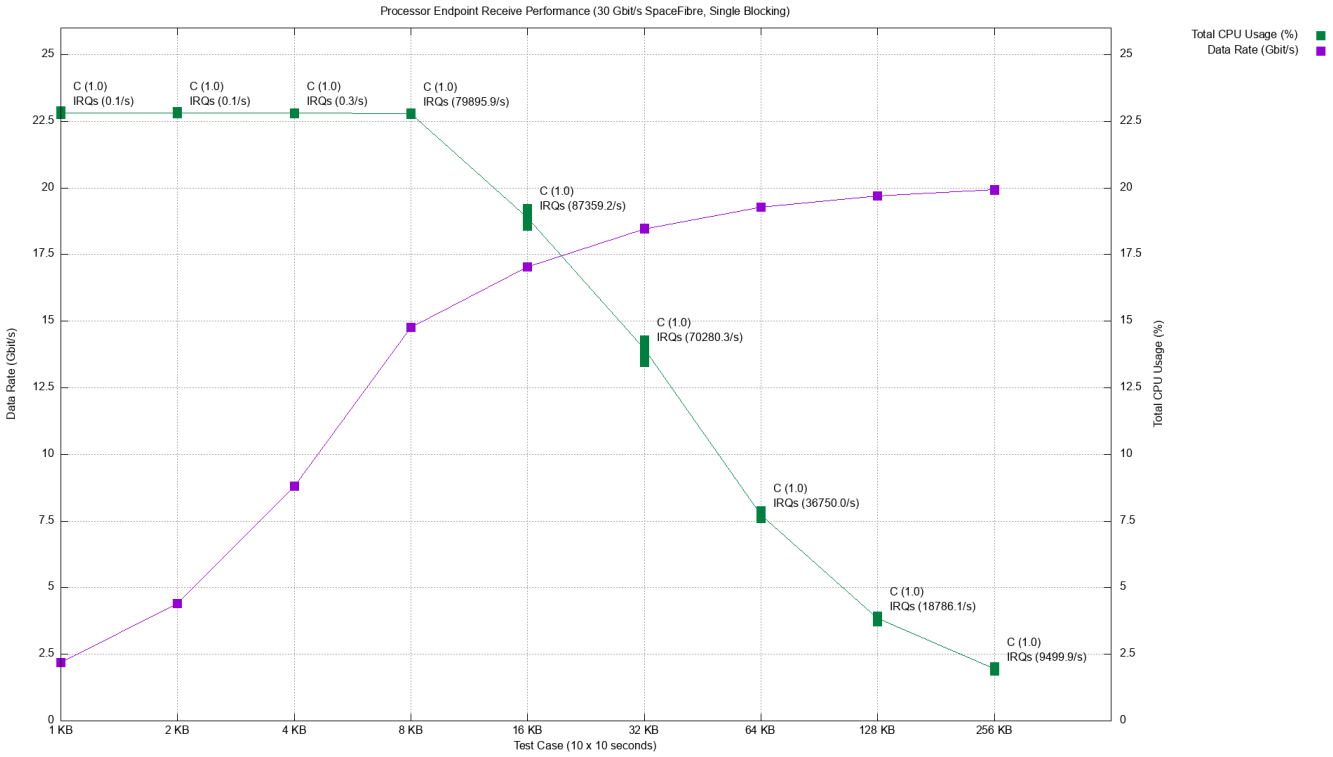

Fig. 7. Receiving Single Packets Chart

As shown in Fig. 7, when receiving single packets, the CPU utilisation is initially high because, similar to sending single packets, there is an overhead to process and clear a completion for each received packet. As the packet length increases, the CPU utilisation reduces, and the data rate increases, because there are fewer, larger packets being received every second.

Similar to sending single packets at a time, the intention of this scenario is to demonstrate that high performance can be achieved even with the simplest approach.

The performance results for receiving single packets at a time are listed in Table III, with the average values across all iterations, followed by the worst-case in parentheses.

TABLE III. RECEIVING SINGLE PACKETS TABLE

Test Case	Data Rate (Gbit/s)	CPU Utilisation (%)
1 KB	2.20 (2.19)	22.82 (22.90)
2 KB	4.40 (4.40)	22.82 (22.86)
4 KB	8.80 (8.79)	22.81 (22.83)
8 KB	14.77 (14.76)	22.79 (22.83)
16 KB	17.04 (17.03)	18.89 (19.23)
32 KB	18.47 (18.46)	13.95 (14.30)
64 KB	19.27 (19.26)	7.72 (7.88)
128 KB	19.69 (19.69)	3.85 (3.95)
256 KB	19.92 (19.91)	1.94 (2.02)

4) Receiving Multiple Packets

To increase the data rate and reduce the CPU utilisation when receiving packets, the completion processing overhead needs to be reduced. This is not as simple as the send side, where packets can be queued and dispatched in groups of N packets, because in a typical application a receiver does not know exactly when or how many packets will arrive so it cannot wait for a fixed number of packets to be received before processing them.

An alternative approach is to wait until the buffers reach a fill threshold, or a timeout occurs, before handling multiple completions at a time. There is a trade-off between reducing latency by processing data as soon as it arrives vs reducing overhead by delaying the processing of data. The SpaceFibre Processor Endpoint software supports multiple polling and blocking options when waiting for operations to complete. For example, the status of an operation can be checked via polling, waited on indefinitely, or waited on with a configurable timeout. This allows a receiver to be tuned appropriately for an application's traffic patterns and requirements.

The performance results for receiving multiple packets, using the threshold filling approach, are illustrated in Fig. 8.

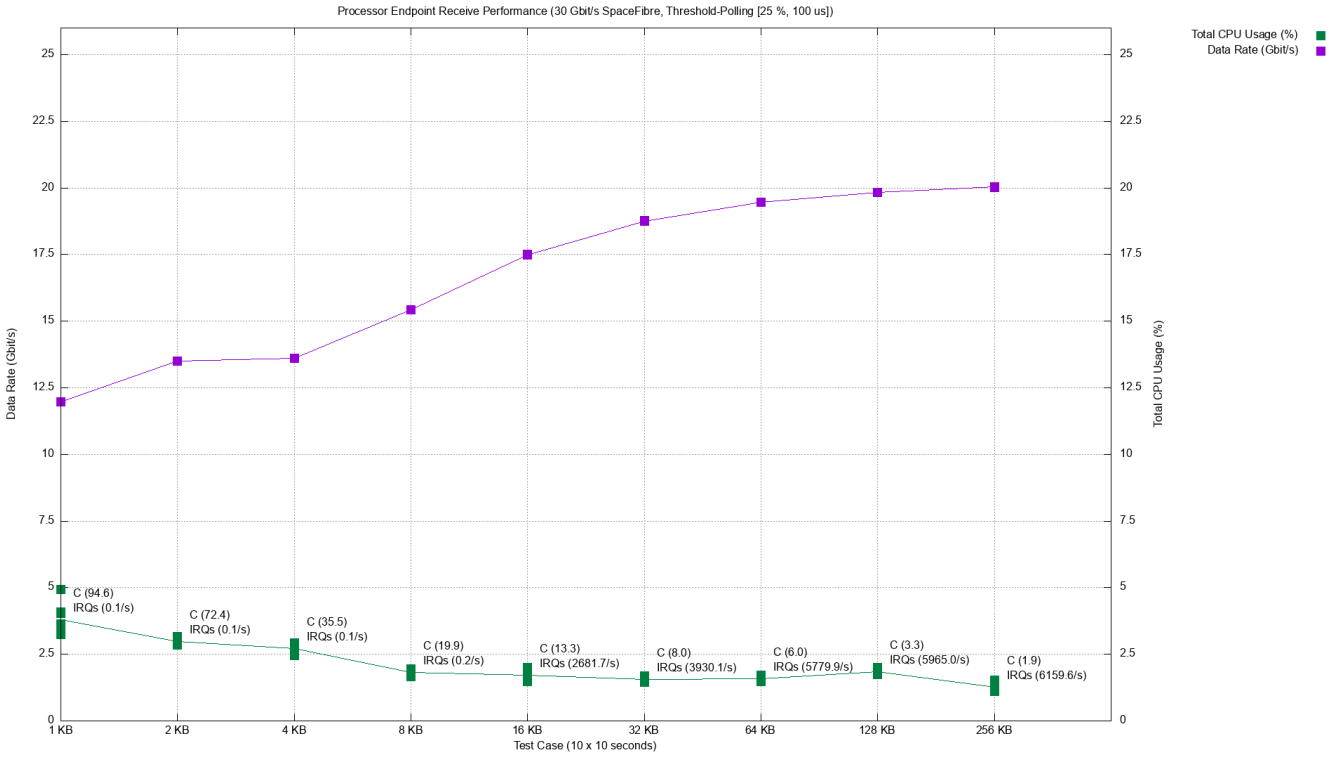

Fig. 8. Receiving Multiple Packets Chart

As shown in Fig. 8, the data rate and CPU utilisation results starting at 1 KB packets are significantly better than the results in Fig. 7. The CPU utilisation continues to be lower as the packet size increases, until the data rate saturates at approximately 20 Gbit/s.

In this scenario, the threshold was set to 25% and the timeout was set to 100 µs, meaning that a receive operation would wait until either the buffer was 25% full or a 100 µs timeout had passed after a newly received packet before handling the current set of receive completions.

The performance results for receiving multiple packets, using the threshold filling approach, are listed in Table IV, with the average values across all iterations, followed by the worst-case in parentheses.

TABLE IV. RECEIVING MULTIPLE PACKETS TABLE

Test Case	Data Rate (Gbit/s)	CPU Utilisation (%)
1 KB	11.96 (11.96)	3.81 (4.92)
2 KB	13.49 (13.48)	2.98 (3.16)
4 KB	13.61 (13.61)	2.72 (2.92)
8 KB	15.42 (15.42)	1.83 (1.95)
16 KB	17.49 (17.48)	1.72 (2.02)
32 KB	18.76 (18.75)	1.56 (1.70)
64 KB	19.46 (19.46)	1.59 (1.72)
128 KB	19.83 (19.82)	1.84 (2.01)
256 KB	20.02 (20.02)	1.27 (1.54)

V. FUTURE WORK

In a recent radiation test, an experimental SpaceFibre link was demonstrated running at 100 Gbit/s in an AMD Versal SoC. To support these extremely high link rates effectively and efficiently in software-based systems, testing of the SpaceFibre Processor Endpoint will be conducted in the near future with AMD Versal SoC devices at a link rate of 100 Gbit/s.

VI. CONCLUSIONS

The SpaceFibre Processor Endpoint has been developed to provide a rapid and simple way to integrate one or more SpaceFibre interfaces into an on-board processing system, such as a SoC, with full software support designed to provide maximum performance with minimal processor overhead.

The SpaceFibre Processor Endpoint consists of an IP core, Linux drivers, and supporting software, providing a hardware-accelerated interface between user-space software applications and a SpaceFibre network. By significantly reducing the work that the processor has to perform to send and receive user data, it provides maximum performance with minimal processor overhead.

The performance results presented in this paper included the data rate and CPU utilisation figures for various sending and receiving scenarios for packets of increasing sizes. The results demonstrated that high performance can be achieved even with the simplest use of the software, and performance can then be significantly improved using straightforward optimisations.

REFERENCES

[1] 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.

[2] S. Parkes, A. Ferrer, A. Gonzalez and D. Gibson, “SpaceFibre Onboard Interconnect: from Standard, through Demonstration, to

Space Flight", IEEE Aerospace Conference, Big Sky, USA, March 2025.

[3] ECSS Standard ECSS-E-ST-50-12C Rev.1, "SpaceWire, Links, Nodes, Routers and Networks", Issue 3, European Cooperation for Space Data Standardization, May 2019, available from http://www.ecss.nl.

[4] S. Parkes and P. Armbruster, "SpaceWire: a spacecraft onboard network for real-time communications", Proceedings of the 14th IEEE-NPSS Real Time Conference, Stockholm, Sweden, June 2005.

[5] A. Gonzalez Villafranca, A. Ferrer Florit, M. Farras Casas and S. Parkes, "SpaceFibre IP Cores for Fast Adoption of Next-Gen FPGA Communication Architectures", IEEE Aerospace Conference, Big Sky, USA, March 2025.

[6] AMD, "AMD Versal Adaptive SoCs", https://www.amd.com/en/products/adaptive-socs-and-fpgas/versal.html

[7] D. Gibson, S. Mills, A. MacLennan and S. Parkes, "Efficient High Data Rate Networking Using Remote Direct Memory Access Over SpaceFibre", European Data Handling & Data Processing Conference (EDHPC), Juan Les Pins, France, October 2023.

[8] AMD, "Zynq UltraScale+ MPSoCs", https://www.amd.com/en/products/adaptive-socs-and-fpgas/soc/zynq-ultrascale-plus-mpsoc.html

[9] Microchip, "PolarFire SoC FPGAs", https://www.microchip.com/en-us/products/fpgas-and-plds/system-on-chip-fpgas/polarfire-soc-fpgas

[10] Microchip, "PolarFire FPGA Evaluation Kit", Microchip, "PolarFire SoC FPGAs", https://www.microchip.com/en-us/development-tool/mpf300-eval-kit

[11] S. Parkes, D. Gibson, S. Mudie, C. McClements, A. Ferrer Florit and A. Gonzalez Villafranca, "Testing SpaceFibre Interfaces and Systems", Data Systems in Aerospace (DASIA) Conference, Online, September 2021.

[12] 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.