Development of UDP based data-streamer and slow-control protocol for streaming readout FEE in SPADI alliance

Ryotaro Honda, Hidetada Baba, Yuto Ichinohe, and Yun-Tsung Lai

***Abstract*—The Signal Processing and Data Acquisition Infrastructure (SPADI) alliance in Japan promotes the development of streaming readout DAQ systems utilizing Ethernet as the communication protocol for front-end electronics (FEE). While the SiTCP, which is widely used in Japan, provides reliable TCP-based data transfer and UDP-based slow control, its reliance on TCP imposes CPU overhead on host PCs. In addition, SiTCP lacks support for simultaneous multi-host slow control access and dynamic IP configuration.**

To address these limitations, we are developing a new Ethernet communication protocol for FEE enabling high-throughput, low-overhead UDP-based data streaming and flexible slow control. The adoption of UDP eliminates the need for the TX buffer used for retransmissions in TCP, improving the memory usage of FEE. Additionally, it frees FEE from the backpressure imposed by PCs during congestion.

The proposed protocol is based on the MAC, IPv4, and UDP layers, which have been developed within the IDROGEN project in France. On top of these layers, we developed a data streamer and a slow control protocol. The data streamer efficiently encapsulates any format of DAQ data in UDP packets and supports multiple independent UDP streams as well as bidirectional communication. The slow control block provides register access through a simplified internal bus with an AXI4-Lite interface. This block has a packet buffer allowing access from multiple hosts.

***Index Terms*—Ethernet, UDP, FPGA, SlowControl**

I. Introduction

The Signal Processing and Data Acquisition Infrastructure (SPADI) alliance has been formed [1], working toward the development and standardization of streaming readout data-acquisition (DAQ) system in Japan. In the SPADI alliance, the Ethernet based data communication has been adopted for the data link of front-end electronics (FEE) to utilize Commercial Off-The-Shelf (COTS) components. The slicon TCP (SiTCP) is proprietary communication protocol for field programmable gate array (FPGA) widely used in particle and nuclear experiments in Japan [2]. SiTCP provides fast and reliable data transmission through Transmission Control Protocol (TCP) and addressed-register control based on the proprietary packed formed on the User Datagram Protocol (UDP) payload. Currently, SiTCP for 1GbE and SiTCP-XG [3] for 10GbE are available.

While SiTCP ensures reliable data transmission via TCP, PCs must allocate significant CPU resources for data reception. This makes it less than ideal for streaming readout DAQ systems, where data transfer volumes from FEE tend to be large. Furthermore, SiTCP has several drawbacks listed as follows.

- Simultaneous addressed-register control requests from multiple hosts
- Dynamic IP configuration
- Data streaming to/from multiple destinations
- Use of jumbo frame

To overcom these issues, we have developed a new communication protocol for FEE that retains the strengths of SiTCP while incorporating UDP-based data streaming functionality and addressed-register control capabilities. In recent years, various studies have explored methods for transmitting data from circuits without packet loss using UDP, and know-how for lossless communication up to 100 Gbps already exists [**?**]. The adoption of UDP eliminates the need for the TX buffer used for retransmissions in TCP, improving the memory usage of FEE, which has limited memory resources. Additionally, it frees FEE from the backpressure imposed by PCs during congestion, enhancing FEE's independence from PCs.

II. Design Of Protocol

Adaptive Network Messaging for Open Instruments (ANEMOI) protocol has been developed over the UDP layer. The entire architecture inside the FPGA is illustrated in Fig. 1. This research is based on the results developed by IJCLab in France. We directly reuse the MAC layer, IPv4 layer, and UDP layer blocks implemented on their IDROGEN board, which is currently under development. These blocks handle responses to DHCP and ping (ICMP Echo Request).

The ANEMOI core consists of the addressed-register control (ANEMOI ARC) block and the data streamer block, their interface to the UDP layer is AXI4-Stream. The ANEMOI ARC block is designed to be adaptive to any Ethernet speed thanks to the AXI4-Stream width converter. This block provides a slow control bus to user circuits in either AXI4-Lite format or a proprietary bus format where the bus address and data width are 32-bit. The user can choose which format to

This work is supported by JST K-program, Grant number of JPMJKP24J2. We would like to acknowledge the members of the IDROGEN project for kindly sharing the source code for the UDP framework. We would like to thank the technical support from the members of the Signal Processing and Data Acquisition Infrastructure (SPADI) alliance.

Ryotaro Honda and Yun-Tsung Lai are with Institute of Particle and Nuclear Studies, High Energy Accelerator Research Organization, Tsukuba 305-0801, Japan (e-mail: rhonda@post.kek.jp; ytlai@post.kek.jp).

Hidetada Baba and Yuto Ichinohe are with RIKEN Nishina Center, Hirosawa, Wako, Saitama 351-0198, Japan (e-mail: baba@ribf.riken.jp; ichinohe@ribf.riken.jp)

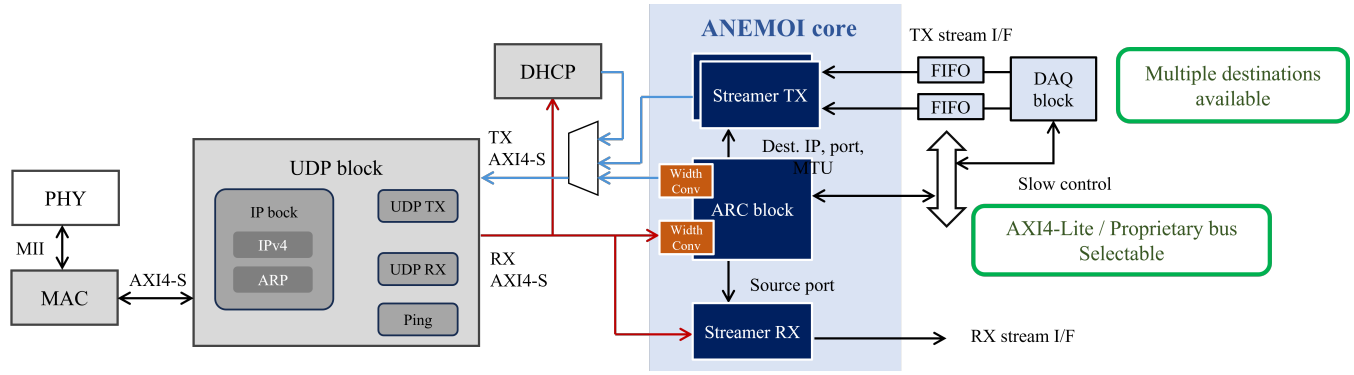

Fig. 1. Block diagram of the ANEMOI protocol. The MAC, IP, and UDP blocks are from the IDROGEN project. The AXI4-Stream format is adopted as the interface to the UDP layer. Users can instantiate multiple Streamer TX blocks resulting in multiple destinations. The addressed-register control (ARC) block the slow control bus with either the AXI4-Lite format or proprietary format. The ARC block configures the destination IP address, the port number, and the MTU value for the Streamer TX.

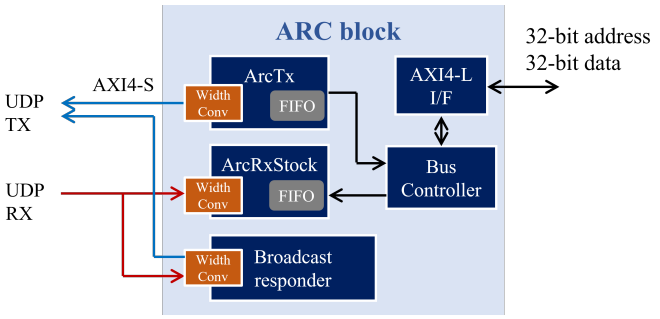

Fig. 2. Block diagram of ANEMOI ARC block. The ArcRxStock buffers the ARC packet and parse it. The BusController manages the bus access. The ACK packet is generated according to the access result and is sent from the ArcTx block to the host PC. The broadcast responder returns the packet containing the MAC address, the IP address, and pre-programmed user value in response to the limited broadcast with specif format.

use. The Streamer TX provides high-speed data transmission to PCs and other FEEs. The Streamer hides the Ethernet frame structure; by implementing a FIFO in front of the Streamer interface, users can transmit data without having to be aware of Ethernet frame sizes. Users can configure multiple destinations by instantiating multiple TX blocks. The destination IP address, port number, and MTU value can be configured via the ANEMOI ARC. The Streamer RX interprets UDP payloads sent in the ANEMOI Streamer format and forwards the data stream to the user circuitry.

A. ARC Block

The internal architecture of the ARC block is represented in Fig. 2. To accept access from multiple host PCs, packets sent to the specific port number are temporarily buffered in the ArcRxStock unit. The packets are interpreted in sequence, and the results are passed to the BusController. The BusController drives the slow control bus to read from and write to the user circuit. Based on these results, the ArcTx unit generates an ACK packet and sends it back to the host PC.

Packets consist of 32-bit data, with a header area of 64 bits. The header, starting from the most significant bit, consists of an 8-bit magic word, a 4-bit version, a 4-bit command, an 8-bit mode, an 8-bit flag, a 16-bit length, and a 16-bit area reserved for future use. The address and data are stored following the header. ANEMOI ARC supports sequential mode and list mode. In sequential mode, one address word comes first, followed by N data words. In this mode, the BusController performs multiple accesses to the same address. This is intended for reading from and writing to a FIFO. By enabling an option, the BusController can access the user circuit while incrementing the address; this is intended for sequential access to RAM. In list mode, N pairs of addresses and data words are stored following the header. This mode provides random access to addresses that are all different.

The broadcast responder independently works from the ARC units. It returns the packet containing the MAC address, the IP address, and pre-programmed user value in response to the limited broadcast with specif format

B. Streamer Block

The Streamer TX unit once stores data from user circuit into the internal FIFO and starts transmission when data amount stored in FIFO exceeds the preset packet length or when a certain amount of time has passed after the first data is stored (time out). These parameters are changeable through the ARC block. The user circuit can stream data without considering the structure of Ethernet frame. The present streamer packet is simple, The first 8 bits are the preamble, followed by a 24-bit frame number; the remainder is user data.

The streamer RX contains an internal FIFO and buffers RX data destined for specific port numbers. After removing the preamble and frame number, the data is streamed to the user circuit.

III. Implementation Results

We implemented ANEMOI in the following three modules and conducted communication tests. AMANEQ [4] having XC7K160T-2, UT4 having XCVU080, and CTS having Kria K26 PL block. We also implemented both GbE and 10GbE versions of AMANEQ and performed throughput tests on each. The PC receiving the data is equipped with a Core i9-13900K processor and an FS.com X710BM2-2SP NIC card.

TABLE I
RESULTS OF THROUGHPUT MEASUREMENT.

Data gen. rate (Mbps)	Throughput (Mbps)	Packet loss (%)	MTU (Byte)
GbE			
957.7	957.7	0.00	1500
992.9	992.9	0.00	9200
10GbE			
9140.6	9111.3	0.13	1500
9845.6	9845.4	0.01	9200

The operating system is Ubuntu 22.04.3. The results are summarized in Table I.

In this test, AMANEQ and the PC were directly connected. While no packet loss was observed at 1 Gbps, non-zero loss was confirmed at 10 Gbps. This trend improves when jumbo frames are used.

IV. SUMMARY AND FUTURE PROSPECT

We are developing a UDP-based protocol for future streaming DAQ systems that addresses the issues of existing communication protocols. Named ANEMOI, this protocol consists of an addressed-register control block and a data streamer block, and is designed to be adaptive to various Ethernet speeds. Using the MAC, IP, and UDP blocks provided by the IDOROGE project, we implemented ANEMOI in three types of modules. In throughput measurements using the AMANEQ module, we obtained results of 992.9 Mbps and 9845.4 Mbps for the GbE and 10GbE versions, respectively, using an MTU value of 9200 bytes.

We plan to extend ANEMOI to support 40GbE and 100GbE in the future. As part of this process, we plan to implement forward error correction. ANEMOI is scheduled to be released as open-source software (OSS).

REFERENCES

[1] *SPADI alliance* [Online]. Available: https://spadi-alliance.rcnp.osaka-u.ac.jp/ [In Japanese] Accessed on: July 3, 2026.

[2] T. Uchida, "Hardware-based TCP processor for gigabit Ethernet", *IEEE Trans. Nucl. Sci.*, vol. 55, no. 3, pp. 1631-1637, Jun. 2008. 10.1109/TNS.2008.920264.

[3] *Bee Beans Technologies SiTCP-XG* [Online]. Available: https://github.com/BeeBeansTechnologies/SiTCPXG_Netlist_for_Kintex7 Accessed on: July 3b, 2026.

[4] R. Honda *et al., "General-Purpose Data Streaming FPGA TDC Synchronized by SerDes-Based Clock Synchronization Technique",* IEEE Trans. Nucl. Sci*., vol. 72, no. 3, pp. 614, Mar. 2025. 10.1109/TNS.2025.3541731.*