Nuclear Inst. and Methods in Physics Research, A 1053 (2023) 168339

Contents lists available at ScienceDirect

Nuclear Inst. and Methods in Physics Research, A

journal homepage: www.elsevier.com/locate/nima

Full Length Article

Implementation and performances of the IPbus protocol for the JUNO Large-PMT readout electronics

Riccardo Triozzi [a], Andrea Serafini [a,b,*], Marco Bellato [b], Antonio Bergnoli [b], Matteo Bolognesi [a,b], Riccardo Brugnera [a,b], Vanessa Cerrone [a], Chao Chen [c], Barbara Clerbaux [d], Alberto Coppi [a], Daniele Corti [b], Flavio dal Corso [b], Jianmeng Dong [e], Wei Dou [e], Lei Fan [c], Alberto Garfagnini [a,b], Arsenii Gavrikov [a,b], Guanghua Gong [e], Marco Grassi [a,b], Rosa Maria Guizzetti [a], Shuang Hang [d,f], Cong He [c], Jun Hu [c], Roberto Isocrate [b], Beatrice Jelmini [a,b], Xiaolu Ji [c], Xiaoshan Jiang [c,g], Fei Li [c], Zehong Liang [c], Ivano Lippi [b], Hongbang Liu [h], Hongbin Liu [c], Shenghui Liu [c], Xuewei Liu [e], Daibin Luo [c], Ronghua Luo [h], Filippo Marini [a,b], Daniele Mazzaro [b], Luciano Modenese [b], Marta Colomer Molla [d], Zhe Ning [c], Yu Peng [c], Pierre-Alexandre Petitjean [d], Alberto Pitacco [b], Mengyao Qi [c], Loris Ramina [b], Mirco Rampazzo [b], Massimo Rebeschini [b], Mariia Redchuk [b], Yunhua Sun [c], Andrea Triossi [a,b], Fabio Veronese [b], Katharina von Sturm [a,b], Peiliang Wang [c], Peng Wang [d,f], Yangfu Wang [c], Yusheng Wang [c], Yuyi Wang [e], Zheng Wang [c], Ping Wei [h], Jun Weng [e], Shishen Xian [i,j], Xiaochuan Xie [c], Benda Xu [e], Chuang Xu [e], Donglian Xu [i,j], Hai Xu [h], Xiongbo Yan [c,g], Ziyue Yan [c], Fengfan Yang [c], Yan Yang [h], Yifan Yang [d], Mei Ye [c], Tingxuan Zeng [c], Shuihan Zhang [c], Wei Zhang [c], Aiqiang Zhang [e], Bin Zhang [e], Siyao Zhao [h], Changge Zi [c], Sebastiano Aiello [k], Giuseppe Andronico [k], Vito Antonelli [o], Andrea Barresi [p], Davide Basilico [o], Marco Beretta [o], Augusto Brigatti [o], Riccardo Bruno [k], Antonio Budano [q], Barbara Caccianiga [o], Antonio Cammi [r], Stefano Campese [a,b], Davide Chiesa [p], Catia Clementi [s], Marco Cordelli [t], Stefano Dusini [b], Andrea Fabbri [q], Giulietto Felici [t], Federico Ferraro [o], Marco Giulio Giammarchi [o], Cecilia Landini [o], Paolo Lombardi [o], Claudio Lombardo [l,k], Andrea Maino [m,n], Fabio Mantovani [m,n], Stefano Maria Mari [q], Agnese Martini [t], Emanuela Meroni [o], Lino Miramonti [o], Michele Montuschi [m,n], Massimiliano Nastasi [p], Domizia Orestano [q], Fausto Ortica [s], Alessandro Paoloni [t], Sergio Parmeggiano [o], Fabrizio Petrucci [q], Ezio Previtali [p], Gioacchino Ranucci [o], Alessandra Carlotta Re [o], Barbara Ricci [m,n], Aldo Romani [s], Paolo Saggese [o], Simone Sanfilippo [q,1], Chiara Sirignano [a,b], Monica Sisti [p], Luca Stanco [b], Virginia Strati [m,n], Francesco Tortorici [l,k], Cristina Tuvé [l,k], Carlo Venettacci [q], Giuseppe Verde [k], Lucia Votano [t]

[a] *Università di Padova, Dipartimento di Fisica e Astronomia, Padova, Italy*
[b] *INFN Sezione di Padova, Padova, Italy*
[c] *Institute of High Energy Physics, Beijing, China*
[d] *Université Libre de Bruxelles, Brussels, Belgium*
[e] *Tsinghua University, Beijing, China*
[f] *Nanjing University of Aeronautics and Astronautics, Nanjing, China*
[g] *University of Chinese Academy of Sciences, Beijing, China*
[h] *Guangxi University, Nanning, China*
[i] *School of Physics and Astronomy, Shanghai Jiao Tong University, Shanghai, China*
[j] *Tsung-Dao Lee Institute, Shanghai Jiao Tong University, Shanghai, China*
[k] *INFN Sezione di Catania, Catania, Italy*
[l] *Università di Catania, Dipartimento di Fisica e Astronomia, Catania, Italy*

* Corresponding author at: Università di Padova, Dipartimento di Fisica e Astronomia, Padova, Italy.
E-mail address: andrea.serafini@infn.it (A. Serafini).
[1] Now at INFN Laboratori Nazionali del Sud, Italy.

https://doi.org/10.1016/j.nima.2023.168339
Received 5 March 2023; Received in revised form 21 April 2023; Accepted 2 May 2023
Available online xxxx

[m] INFN Sezione di Ferrara, Ferrara, Italy
[n] Università degli Studi di Ferrara, Dipartimento di Fisica e Scienze della Terra, Italy
[o] INFN Sezione di Milano e Università di Milano, Dipartimento di Fisica, Milano, Italy
[p] INFN Sezione di Milano Bicocca, e Università di Milano Bicocca, Dipartimento di Fisica, Milano, Italy
[q] INFN Sezione di Roma Tre e Università di Roma Tre, Dipartimento di Matematica e Fisica, Roma, Italy
[r] INFN, Sezione di Milano Bicocca e Politecnico di Milano, Dipartimento di Energetica, Milano, Italy
[s] INFN Sezione di Perugia e Università di Perugia, Dipartimento di Chimica, Biologia e Biotecnologie, Perugia, Italy
[t] Laboratori Nazionali dell'INFN di Frascati, Italy

ARTICLE INFO

Keywords:
Electronics
Photomultiplier
Large scale neutrino experiment

ABSTRACT

The Jiangmen Underground Neutrino Observatory (JUNO) is a large neutrino detector currently under construction in China. Thanks to the tight requirements on its optical and radio-purity properties, it will be able to perform leading measurements detecting terrestrial and astrophysical neutrinos in a wide energy range from tens of keV to hundreds of MeV. A key requirement for the success of the experiment is an unprecedented 3% energy resolution, guaranteed by its large active mass (20 ktons) and the use of more than 20,000 20-inch photo-multiplier tubes (PMTs) acquired by high-speed, high-resolution sampling electronics located very close to the PMTs. As the Front-End and Read-Out electronics is expected to continuously run underwater for 30 years, a reliable readout acquisition system capable of handling the timestamped data stream coming from the Large-PMTs and permitting to simultaneously monitor and operate remotely the inaccessible electronics had to be developed. In this contribution, the firmware and hardware implementation of the IPbus based readout protocol will be presented, together with the performances measured on final modules during the mass production of the electronics.

1. Introduction

The Jiangmen Underground Neutrino Observatory (JUNO) is a 20-kton multi purpose underground liquid scintillator detector currently under construction in the Guangdong Province in South China [1]. The main goal of the JUNO experiment is the determination of the Neutrino Mass Ordering (NMO), which will be resolved by analyzing the oscillation pattern of the electron anti-neutrinos produced in the nuclear fissions of the 52.5-km-distant Yangjian and Taishan Nuclear Power Plants. Within the first six year of data-taking, JUNO is expected to determine the NMO at a 3σ significance [1], simultaneously measuring the Δm^2_{31}, Δm^2_{21}, $\sin^2\theta_{12}$ oscillation parameters to a sub percent World leading precision [2]. Besides its main ambitious goal, JUNO's extensive physics program includes studies of neutrinos from the Sun, the atmosphere, supernovae, and planet Earth, as well as explorations of physics beyond the Standard Model [1].

In order to fulfill its ambitious goals, JUNO aims to achieve a better than 1% energy linearity and a 3% energy resolution at 1 MeV [3]. This unprecedented resolution will be reached by featuring a dual calorimetry system [4] consisting of 17,612 20-inch Large-PMTs [5] and a secondary 25,600 3-inch Small-PMTs system [6] in the central detector, resulting in a remarkable 78% photon sensor coverage [1]. Additionally, 2400 20-inch Large-PMTs will be installed in the 35-kton water pool surrounding the detector, acting as part of the muon veto system. While for Small-PMTs only the temporal and deposited charge information will be gathered for each signal, for Large-PMTs the entire waveforms will be collected and stored, resulting in a challenging amount of data to manage. In JUNO, the design trigger rate during data-taking is 1 kHz [1]; this value has been used as target for the development of the readout electronics. A reliable readout acquisition system capable of handling the synchronized data stream coming from the Large-PMTs is a strict requirement for the success of the experiment.

The signal acquired by the 20,012 Large-PMTs used in the central detector and the water tank will be digitized by 7000 Global Control Unit (GCU) boards. With the aim of ensuring minimum electrical noise, the GCUs will be installed underwater in the vicinity of the PMTs. After the filling of the experiment and during data taking, the GCU boards will be not accessible anymore. It is therefore crucial to design a protocol permitting to monitor and operate remotely the electronic boards for the entire duration of the experiment. These requirements are addressed through the development of a custom designed readout electronics [7] and the implementation of an IPbus-based protocol [8] for the simultaneous transmission of data and the slow control of the GCU boards' parameters through Ethernet. In order to test the performances of the IPbus implementation in the JUNO Data AcQuisition (DAQ) streams, extensive rate and bandwidth measurements were collected using two different facilities. Results are reported in the following sections.

2. Setup overview

2.1. JUNO large PMT electronics

The JUNO electronics chain is composed of (i) the *front-end* (FE), or *wet*, electronics [9] submerged in the water pool and (ii) the *dry* electronics, installed in a dedicated electronics room of the JUNO underground laboratories, which consists of the *back-end* (BE), or trigger, electronics and the DAQ system. The FE and the BE communicate through a *synchronous link* established via CAT6 cables, while data is transferred to the DAQ system via an *asynchronous link* running on CAT5 cables. A scheme of the JUNO large-PMT electronics is provided in Fig. 1. The design is an optimization of the one proposed in previous developments [7].

The FE will be installed on the JUNO Steel Truss structure, inside a stainless steel, water-tight box, the so-called Under Water Box (UWbox) and comprises (i) a PMT Voltage Divider located inside the PMT shell, (ii) a High Voltage Unit providing the bias voltage to the voltage divider and (iii) a GCU board integrating the Front-End and Read-out electronics, two FPGAs and a large memory buffer given by a 2 GBs DDR3 SDRAM, needed in the exceptional case of a sudden increase of the input rate. The core of the GCU is a main Xilinx Kintex-7 FPGA (XC7K325T), used to manage all the task assigned to the FE (e.g., PMT waveform processing and data readout), supported by an auxiliary Xilinx Spartan-6 FPGA (XC6SLX16) which ensures a fail-safe reconfiguration of the GCU by means of a virtual JTAG connection over IPbus. Groups of three PMTs are connected via a 1.5 m coaxial cable to UWBoxes containing the PMT High Voltage generator and the GCUs. The GCUs are responsible for the data digitization, buffering and online processing of the waveforms outputted by each of the three connected PMTs and for the simultaneous monitoring and control of all the relevant parameters (i.e. voltage, temperature).

The BE comprises (i) a Back End Card (BEC) gathering the synchronous signals coming directly from the UWboxes, (ii) a Trigger and Timing Interface Mezzanine (TTIM) located within the BEC and

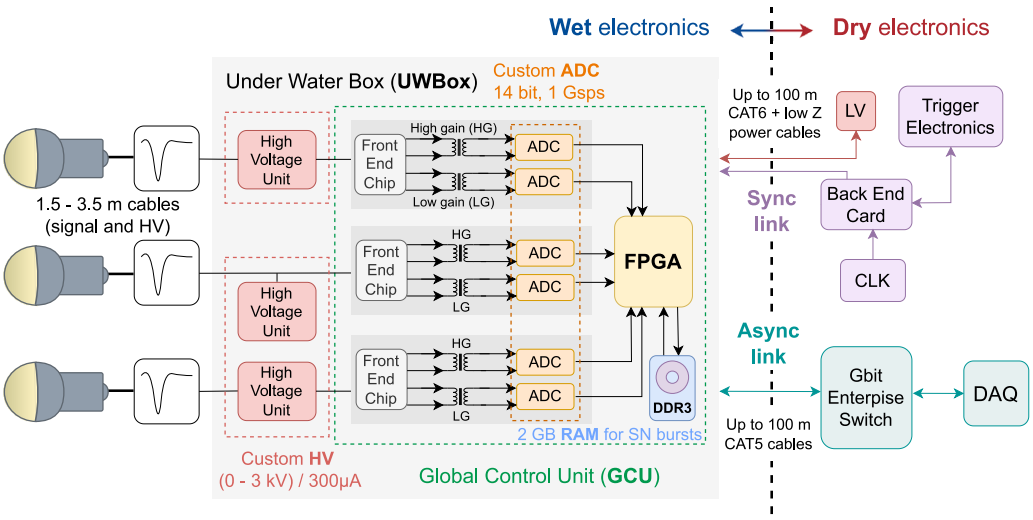

Fig. 1. JUNO large PMT electronics Read-Out electronics scheme. A description of the different parts is given in the text.

representing the main component of the experiment trigger and timing system, (iii) a Reorganize and Multiplex Unit (RMU) providing a front-end trigger optical bridge and (iv) a Central Trigger Unit (CTU) distributing trigger information and global time to the RMUs and consequently to the GCUs.

Groups of 48 GCUs are connected through the *synchronous link* to the TTIM FPGA Mezzanine Card (FMC) of a BEC. The TTIM is then connected to the RMU, which interfaces with 20 different BECs. Finally, each RMU connects to the CTU, that generates a trigger validation and sends it back to all the GCUs involved in a collected event. All the acquired information is transferred through the Gbit Ethernet *asynchronous link* to a Gbit Enterprise Switch and finally to the DAQ system (via 10-Gbit fiber-optic cables) by means of the IPBus Core protocol [8].

A more detailed review of the JUNO large PMTs Read-Out electronics chain can be found in [10].

2.2. The JUNO test facilities

To test the large PMTs Read-Out electronics and the IPbus performances in a realistic scenario, two different setups were assembled. A small JUNO mock-up system exploiting the full electronics chain from the PMTs to the DAQ server was set up at the Legnaro National Laboratories (LNL)[2] of the Italian National Institute for Nuclear Physics (INFN) [11,12] in Legnaro (Padua, Italy). This allowed to test the correct interplay among the different read-out components and the flawless implementation of the IPbus protocol. A second apparatus was set up in a dedicated test facility in Kunshan (China), where the mass production and assembling of the GCUs is currently underway [13,14]. The large-scale setup built in Kunshan allowed to evaluate extensively and realistically the electronics performances on a scale comparable to that envisaged for the JUNO experiment.

LNL setup. The medium size setup installed at LNL features 48 PMTs, associated with as many independent channels. The 2-inch PMTs[3] are submerged in a cylindrical container holding 17 liters of liquid scintillator. A total of 16 GCUs are connected to a single BEC conveying the synchronous information to an RMU coupled to a CTU. The DAQ system is installed in a dedicated server running Ubuntu 18, featuring 32 GB of RAM and 2x Intel Xeon Silver 2.2 GHz CPUs, for a total of 20 cores and 40 threads. An extensive and technical description of the apparatus can be found in [10,11].

Kunshan setup. The Kunshan test facility hosts a large-scale setup where a total of 344 GCUs can be managed simultaneously. Groups of 40 GCUs are handled by a total of 9 BECs, connected to as many RMUs controlled by a single CTU. The DAQ software is installed on a Intel Xeon Gold based server running on CentOS7, featured with 192 GB of RAM and, 2x 2.7GHz processors, for a total of 24 cores and 48 threads. A detailed description of the Kunshan test facility can be found in [13].

3. IPbus implementation in the data acquisition stream

One of the main challenges for the JUNO Large-PMTs Read-Out chain is the possibility to acquire and transfer data in parallel to the remote monitoring and control of the electronics. After the filling of the JUNO detector, the FE will be inaccessible for the entire duration of the data-taking (expected to run for 30 years [1]). It is therefore imperative to implement a protocol assuring a reliable transmission of data, while allowing the asynchronous request and remote modification of acquisition parameters. In this respect, the IPbus suite of software and firmware implements a reliable high-performance control link specifically suited for particle physics electronics [8], as demonstrated by its successful employment, among others, in the CMS [15], ATLAS [16] and ALICE [17] experiments. This control link enables read/write operations on memory-mapped resources over a virtual A32/D32 bus implemented in IP-aware hardware devices.

IPbus is a hardware and firmware solution that communicates over Ethernet using UDP/IP. It consists of (i) a firmware module implementing the IPbus protocol within end-user hardware (e.g. JUNO's Kintex-7 and Spartan-6 FPGAs), (ii) a micro Hardware Access Library (μHAL) providing an end-user C++/Python library for read/write operations on IPbus and (iii) a software application called ControlHub, which mediates simultaneous hardware access from multiple μHAL clients. While the UDP protocol does not include any native reliability mechanism, the use of ControlHub assures the duplication and re-ordering of any lost IPbus UDP packet, providing a reliability mechanism at software level.

3.1. Hardware and firmware implementation

The IPBus protocol is deployed to the GCU via an FPGA core provided in the IPBus suite. This core provides the following modules: (i) a core protocol decoder and (ii) a bus master that controls the interface to several IPBus slaves (e.g. onboard trigger manager, L1-cache, IPbus DAQ, UART-based high voltage unit, I^2C-based temperature sensors, DDR3 memory). Each slave contains a set of registers that can be read and written by IPBus transactions. The addresses, sub-addresses and

[2] Laboratori Nazionali di Legnaro, Viale dell'Università, 2 - I-35020 Legnaro (Padua), Italy.

[3] Philips XP2020 photomultiplier tubes.

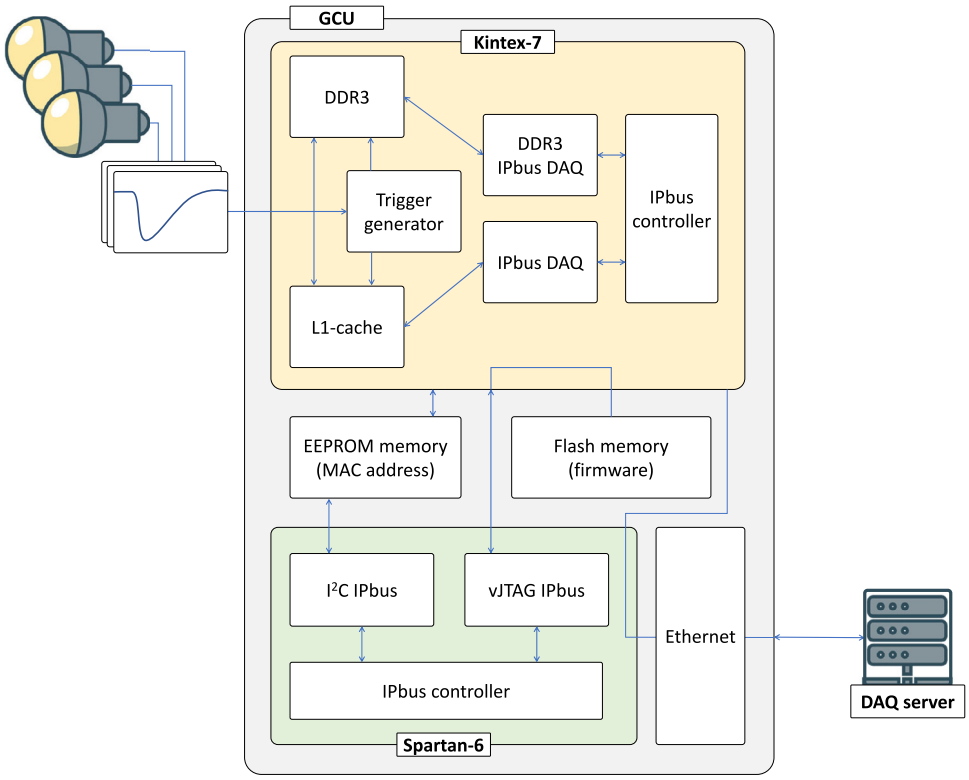

Fig. 2. Scheme of the IPbus implementation within the GCU. The tasks and cores of the Kintex-7 (in yellow) and Spartan-6 (in green) FPGAs are indicated in the white boxes.

masks of these slaves registers can be specified by Extensible Markup Language (XML) files.

In the Kintex7 FPGA, the IPbus master–slave interface is mainly used for data acquisition (see Fig. 2). Once that an analog waveform coming from a PMT has been digitized by the ADC, the digital data follows a three-fold path. The data is sent to (i) a circular buffer called L1-cache capable of storing 32 μs of raw data per channel, (ii) a 2 GB DDR3 memory accommodating high trigger rate scenarios and (iii) a trigger algorithm which extracts the timing information and sends the obtained timestamp to the BE, the DDR3 packager and the L1-cache.

The DDR3 memory is designed to run in self triggering mode (no validation from the CTU is required). This configuration is designed to avoid possible data loss during high-rate events in case of nearby supernovae bursts. The large 2 GB DDR3 memory mounted on the Kintex-7 board is dimensioned to hold more than one million events per channel, making it possible, for example, to capture more than 10 seconds of waveforms for a supernova explosion occurring at 10 kpc. [1]. After receiving the correct timestamp, a packager module encapsulates the digitized event within an header and a trailer and sends it to a DDR3 controller module which addresses the memory as a circular buffer. Whenever the DDR3 content is requested by the DAQ via an IPbus transaction, the writing operations are blocked and the DDR3 is emptied out by the IPbus DDR3 DAQ module. Once the whole content is read, the writing restarts automatically. For rare supernovae bursts, standard L1-cache DAQ will be stopped and the entire available bandwidth will be devoted to the copy and transfer of the DDR3 memory content to the DAQ server. After all of the supernova events will be transferred and stored, standard acquisition via the L1-cache will be restarted.

The L1-cache is instead designed to work in external triggering mode (trigger requests needs to be validated by the CTU), but can be programmed to work in self triggering mode via IPbus if necessary. As soon as a timestamped trigger arrives to the L1-cache, a fixed data payload is extracted from the buffer, packetized within an header and trailer, and sent to the IPbus DAQ module. The IPBus DAQ module presents a "funnel" asynchronous First In, First Out (FIFO) buffer of 2^{13} bytes able to contain four waveform packets, which can be read via IPbus transactions. The IPbus controller core, that provides the interfaces for the IPbus DAQ and IPbus DDR3 DAQ slaves, encapsulates the payload data into UDP network compliant packets and sends the data to the server via the Ethernet MAC module.

In the Spartan-6 FPGA, the IPbus master–slave interface is instead mainly used for the MAC address assignment and for permitting the remote reprogramming of the Kintex-7's firmware (Fig. 2). An I^2C IPbus slave is used for writing/reading a small EEPROM memory storing the MAC address. A dedicated on-board bus then transfers the MAC address information from the Spartan-6 to the Kintex-7 FPGAs. A virtual JTAG (vJTAG) IPBus slave is instead used to expose the Kintex-7 JTAG hardware access to the network. The vJTAG IPbus slave receives from the server the JTAG commands encapsulated in a standard network transport layer, extracts the instructions and then uses a dedicated bus to exchange the JTAG commands to/from the Kintex-7.

3.2. Software implementation

The server-side implementation of the IPbus is divided into three parts: (i) a high-performance C++ μHAL client for the acquisition of data, (ii) a Python μHAL client for the low-frequency monitoring of the GCU slow control parameters and (iii) a μHAL client for the upgrade of the GCUs' firmware and the eventual debug of firmware errors. All three of these data streams pass through the ControlHub server to reach the GCUs (Fig. 3). The communication between the μHAL clients and the ControlHub occurs through the TCP/IP protocol, meaning that components can reside on the same machine or in different machines if necessary. During the mass testing, components operated on the same server, but they are envisaged to be deployed to different machines in the JUNO experiment.

Firmware flashing. The Vivado Lab software suite [18] is used to reconfigure and update the firmware of the Xilinx Kintex-7 boards. This software is used to manage and program the FPGA boards through a

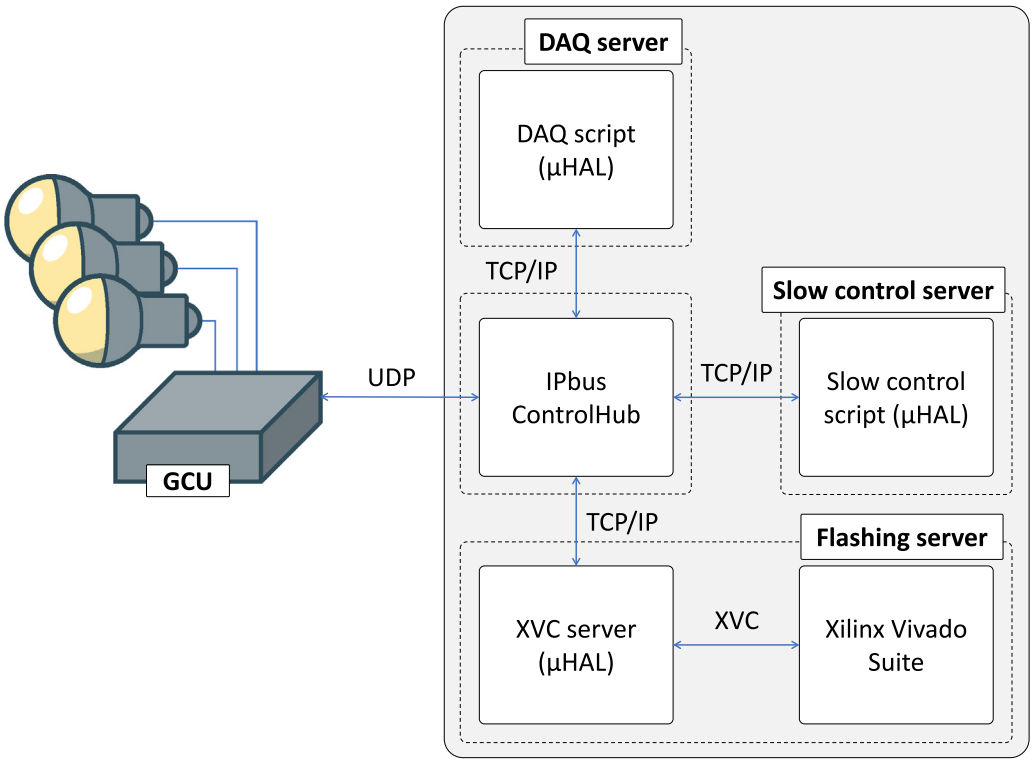

Fig. 3. Scheme of the IPbus implementation within the DAQ infrastructure. The μHAL clients serving the DAQ, slow control and reprogramming capabilities communicate through the TCP/IP protocol with the ControlHub, which manages the requests to/from the GCUs via UDP. A custom Xilinx Virtual Cable (XVC) is used to route the Vivado Lab software connection to the GCU via Ethernet. The three μHAL components can operate within the same machine or in different machines, communicating via TCP/IP over Ethernet.

JTAG connection, a common hardware interface that provides a way to communicate directly with the chips on a board. This connection is also used to monitor debugging parameters pre-established during firmware development (e.g. number of bit-flips in the sync link, IPbus commands sent to the GCU). Since the only connection between the server and the GCUs is an Ethernet link, we rely on the TCP/IP-based Xilinx Virtual Cable (XVC) protocol to access and debug the FPGA over Ethernet. A custom XVC daemon installed on the server receives the requests from the Vivado Lab software and routes them to the GCUs via IPbus through the ControlHub.

The Kintex-7 reprogramming procedure consists of two steps. First a *programmer* is mounted on the volatile memory of the FPGA. This is a firmware that pings on a specific IP address and which will be used to write the firmware on the permanent flash memory of the Kintex-7. Once installed the *programmer*, the FPGAprog utility [19] is used to write the new firmware to the board through the specific *programmer*'s IP address. While the flashing of the *programmer* has a great impact on the server resources, the programming of the permanent flash memory is extremely lightweight.

At the end of the procedure, the FPGA reboots automatically and loads the updated firmware from the flash memory. For a single GCU this procedure takes a few seconds when happening through a physical JTAG link, but can take up to 2 min when relying on a virtual JTAG connection over Ethernet. Since the JUNO experiment will employ more than 7000 GCUs, it was necessary to develop a procedure for a parallel and efficient reprogramming of the cards.

A workers-based system was therefore developed in bash.[4] A number of *workers* equal to the number of available cores are initialized (i.e. 24 in our server). A *job* is assigned to each request for programming a GCU and is inserted in a FIFO queue. As soon as a *worker* is available, it takes the first *job* from the FIFO to execute it. When the *worker* finishes the *programmer*'s flashing *job*, it launches in the background a FPGAprog process for writing the firmware to the GCU's permanent flash memory before moving on to the next *job*. At the end of the procedure, the flashing script verifies that each GCU booted the correct firmware version. In case the GCU firmware update failed, a new *job* is resubmitted to the FIFO queue (see Fig. 4).

Slow control. The IPbus protocol is exploited to access and monitor a variety of internal parameters and sensors present on the GCU and keep track of the board's overall status. The slow control monitoring runs in parallel to the DAQ, sharing the same IPbus transport layer thanks to the ControlHub capabilities. During the slow control, the following parameters are monitored for each GCU: temperature of the FPGA, temperature of each HVU, high voltage value of each HVU, FPGA internal supply voltage, supply voltage for FPGA RAM memories, FPGA auxiliary supply voltage, external reference voltages for FPGA internal ADC.

Data acquisition. For each GCU, a DAQ script based on a C++ μHAL client is initialized for the acquisition of data. The DAQ script queries via an IPbus transaction the IPbus DAQ on board of the GCU. For each request, the script specifies the size of the chunk of data to transfer (i.e. the *DAQ Buffer Size*). Whenever the GCU has enough data to fill the requested buffer, it transfers the data to the server and empties the GCU's FIFO. The FIFO occupancy value parameter is exposed to IPbus, opening up to the possibility of complex adaptive scripts varying the requested block size as a function of the trigger rate and maximizing DAQ efficiency. The *DAQ Buffer Size* (BS) is the length parameter of the memory buffer in which the first chunk of data read by the DAQ is temporarily stored. The BS is expressed in units of 32 bits and its maximum value corresponds to the FIFO depth, that is 2^{13} Bytes (i.e. 2048 32-bit blocks). The acquired data are stored in a binary file for each active channel (3 for each GCU) and are organized in data packets, each one having a variable number of *words*, defined as sequences of 16 bits. Any packet is wrapped by an *header* and a *trailer* 8-words sequences, that provide unique information on each event (Fig. 5).

[4] Bourne Again SHell.

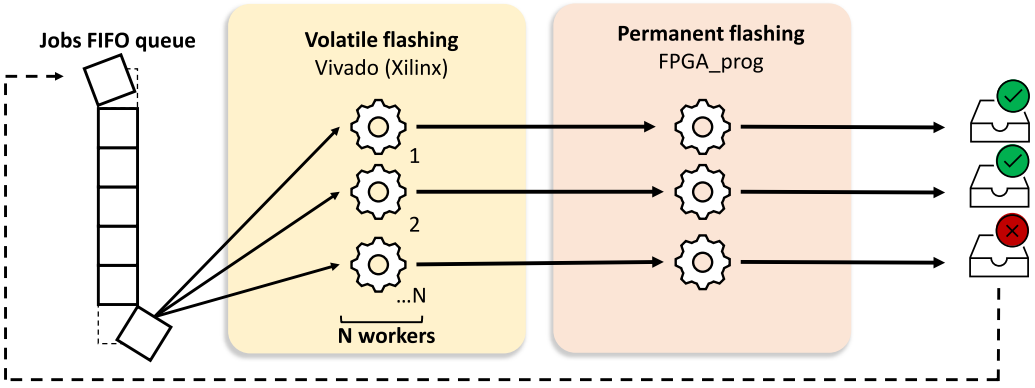

Fig. 4. Illustrative scheme of the procedure adopted for the parallel reprogramming of GCUs.

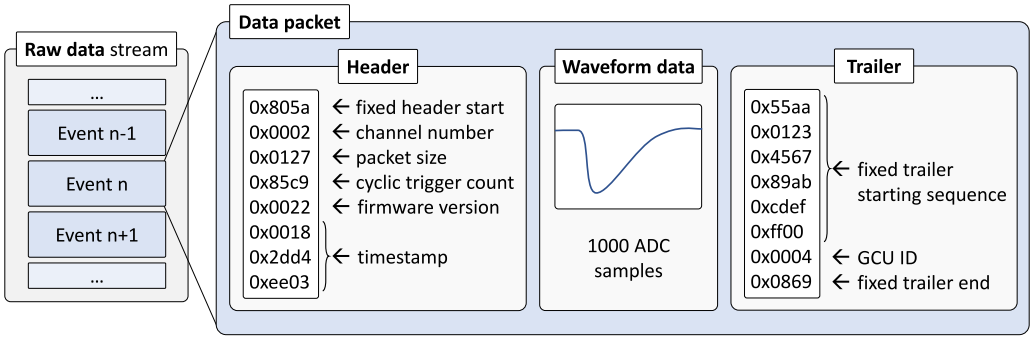

Fig. 5. General structure of the raw data. Every event is composed of header and trailer sequences wrapped around the waveform data.

Apart from the fixed sequences, each header is composed of the GCU channel number, the packet size, the trigger count and the firmware version. The packet size indicates the size of the transferred data packet including header and trailer in units of 8 words. The trigger count is a hexadecimal cyclic number, that can be exploited for debugging. Finally, the last 48 bit sequence is the *timestamp*, which gives the time reference in units of 8 ns. The trailer is composed of a fixed starting sequence, the GCU ID number and a fixed ending word. The actual event data is stored as a sequence of words given by the packet size minus header and trailer and stores the digitized waveform, which starts from the reference timestamp found in the header.

4. IPbus performance assessment

4.1. Firmware flashing

In order to assess the performances of the IPbus based implementation of the reprogramming framework, the time needed to flash a new firmware to a set of 250 GCUs in the Kunshan test setup [13] was measured. The procedure was repeated with different configuration, initializing from time to time a different number of parallel workers capable of handling the flashing jobs. Simultaneously, the server resources allocated to the flashing procedure were monitored (Fig. 6).

The optimal configuration is obtained when initializing a number of workers approximately equal to the number of available cores on the machine (i.e. 24 cores). This configuration yields the minimum time needed to reprogram the complete set of GCUs, while exploiting most of the resources provided by the server. In this configuration it is possible to update the firmware of 250 GCUs in about 15 min.

4.2. Data acquisition

In order to assess the IPbus implementation performances in the data acquisition, several measurements with different trigger rates, different DAQ and network parameters and different number of GCUs run in parallel have been performed. Two different metrics useful for the evaluation of the DAQ performances have been identified: (i) the maximal transferred bandwidth, that is the maximum amount of data transferred from the GCU to the server, and (ii) the survival fraction (or efficiency), that is the number of acquired events over the number of expected triggers. The tests were carried out using an external global trigger with fixed frequency provided by the CTU.

The analysis is carried out on the raw data saved to the server. Data are temporarily stored to ram-disk during acquisition and then moved to disk at the end of the test. For each acquired channel, a dedicated software is used to verify the validity of each event and to extract the timestamp and the metadata of that event. When a channel misses a timestamp or the event data packet is corrupted, the corresponding event is considered lost. The number of acquired events is hence calculated by counting the number of timestamps obtained for each channel. The corresponding bandwidth can be calculated estimating the total amount of data transferred (i.e. the number of events multiplied by the size of each event, 2032 bytes[5]) divided by the acquisition time (i.e. the difference between the last and first timestamps acquired). The number of expected events, on the other hand, is calculated by multiplying the trigger rate by the acquisition time. In this regard, since all tests have been performed by using an external trigger with fixed frequency, the trigger rate is extracted by the timestamp distribution with an arithmetic mean of the timestamp differences between consecutive events. The error attributed to the frequency is derived from the standard deviation of the timestamp differences. The uncertainty on the expected number of events can be obtained by propagating the error on the trigger rate; the error on the number of registered events is estimated to be Poissonian.

[5] 1 waveform packet = 16 header bytes + 16 trailer bytes + 1 μs x 1000 ADC/μs x 2 bytes/ADC = 2032 bytes.

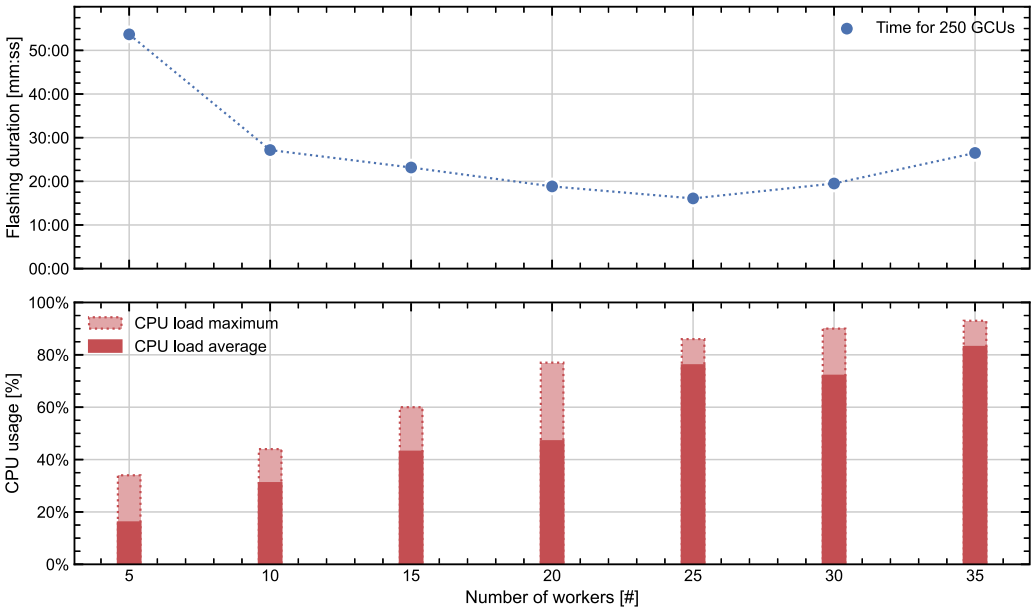

Fig. 6. Duration (top) and corresponding CPU usage log (bottom) for the reprogramming procedure of 250 GCUs.

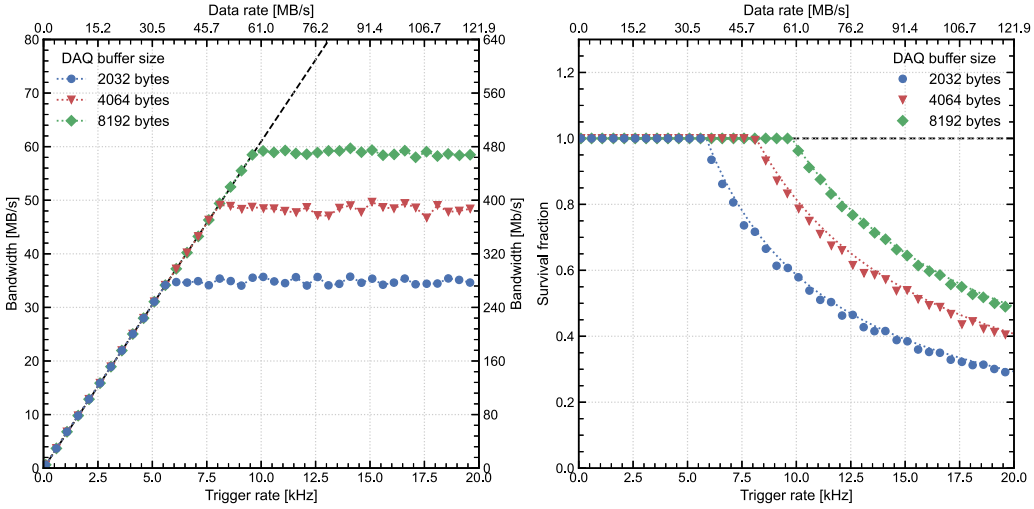

Fig. 7. Transferred bandwidth (left) and corresponding survival fraction (right) for a single GCU as a function of the trigger rate for different DAQ buffer sizes.

The first parameter ruling the maximal performances of the readout is the DAQ buffer size. For a single GCU, the optimal readout performances can be achieved by setting a DAQ buffer size equal to the GCU's onboard IPbus FIFO size. Lower sizes require a higher number of transactions to empty the FIFO, the additional latency limits the reachable maximal bandwidth. Bigger sizes yield higher maximal bandwidths. Once that the maximal bandwidth has been reached, the data readout saturates and part of the acquired events cannot be transferred to the DAQ server (Fig. 7). This permits to reach for a single GCUs readout performances of about 60 MB/s, corresponding to a trigger rate of nearly 10 kHz, well above the design requirements of 1 kHz.

The stability of the readout setup and of the synchronous link has been verified through a 350-h long acquisition performed in the LNL setup [20]. A rate test of cosmic muons has been performed on 39 channels in parallel, using the coincidence of three plastic scintillator bars as an external trigger connected to the BEC. The resulting cosmic muon rate (about 3 Hz) remained stable over almost 14 days on all the 13 employed GCUs for the entire duration of the test, without any clock re-synchronization [21].

When dealing with a higher number of GCUs two possible bottlenecks may be due to (i) network performances and (ii) server resources. For what concerns the network, each GCU accommodates a 1 Gbit/s uplink connected to a 40 Gbit/s switch. In the Kunshan setup, a Terabyte Ethernet port assures transfer bandwidths up to 400 Gbit/s between GCUs and the server itself. This is enough to accommodate for more than 800 GCUs[6] per server reading out at peak bandwidths. Network switch buffering and other secondary effects may prevent the network from reaching its nominal bandwidth. The network has been oversized to ensure that these effects do not pose a problem for acquisition: only groups of 40 GCUs will be connected to each L1 switch so that each L2 switch will only have to handle the data flow of 280 GCUs (Fig. 8), which is less than half of the nominal bandwidth it could manage.

Server resources represent instead a strong constraint and need a thorough investigation. Several acquisitions with an increasing number of GCUs (from 1 to 150) read out in parallel have been performed (Fig. 9). All acquisitions were conducted in the nominal network configuration illustrated in Fig. 8. Simultaneously, the server resources used by the DAQ processes during the acquisitions have been monitored.

[6] Considering that each GCU can output up to 60 MB/s, corresponding to 480 Mbit/s, a 400 Gbit/s port can ideally manage up to ~830 GCUs.

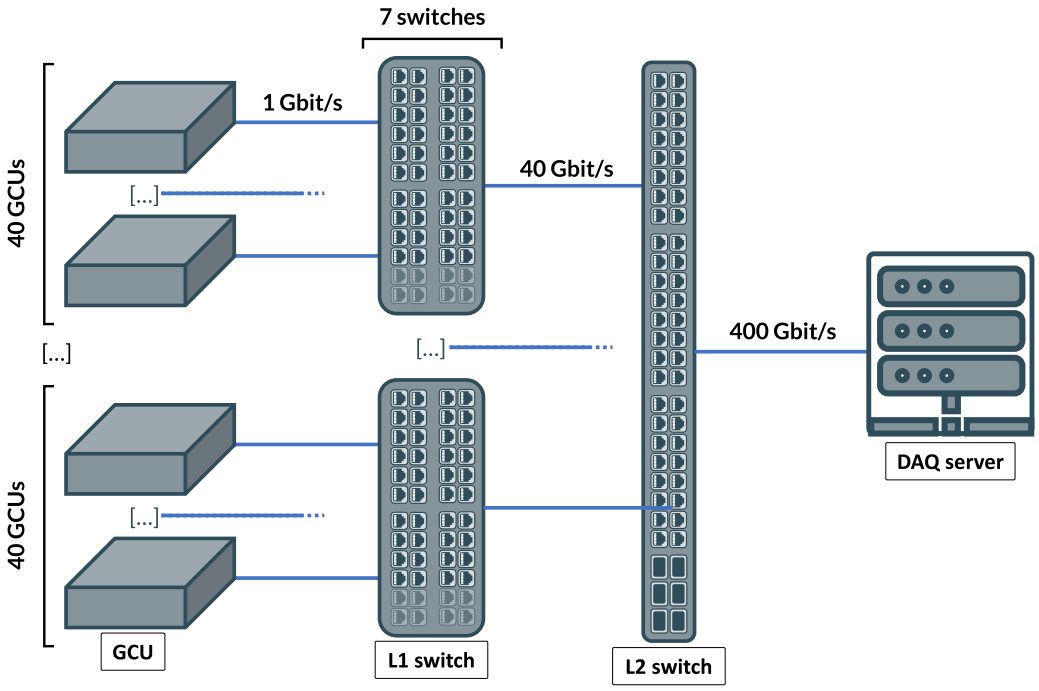

Fig. 8. Scheme of the network infrastructure in the Kunshan setup. Groups of 40 GCUs are connected to a L1 switch. A total of 7 L1 switches are then connected to a single L2 switch communicating with the DAQ server through a Terabyte Ethernet port.

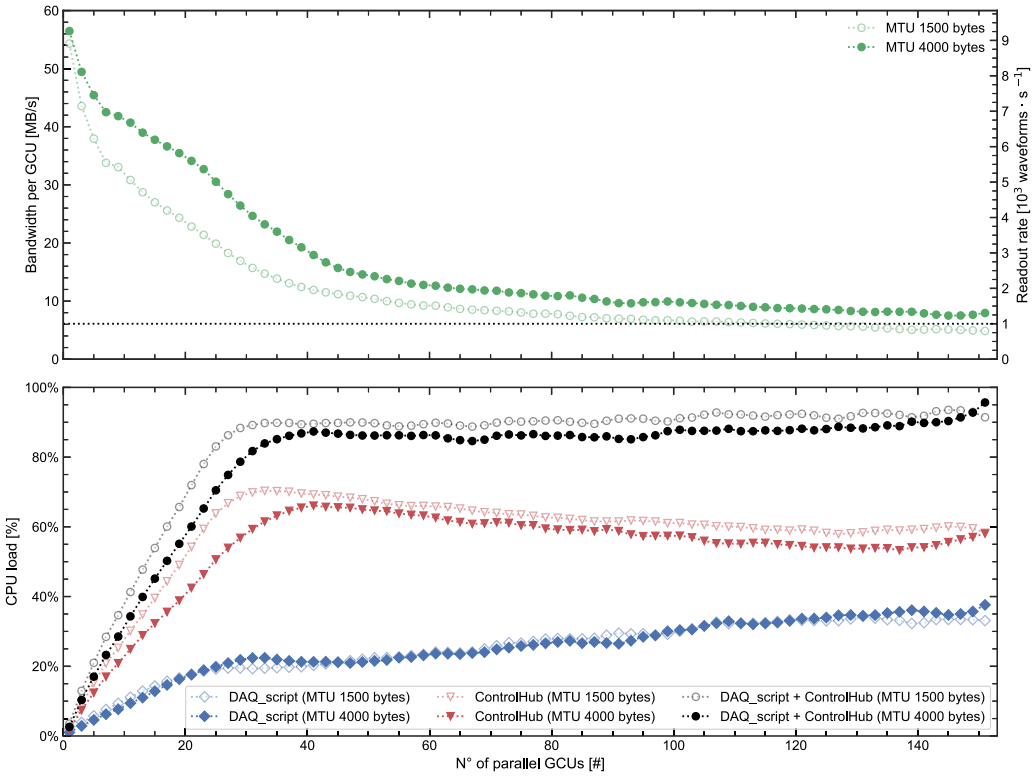

Fig. 9. Relative bandwidth trend (top) and corresponding CPU usage log (bottom) as function of the number of GCUs run in parallel. Each data point corresponds to a 60 s acquisition. The total CPU usage is broken down into its ControlHub and DAQ acquisition script components. Acquisitions have been performed with different Maximum Transmission Units (MTUs) set on the network interfaces, here represented by different levels of transparency.

The CPU usage rapidly increases up to 30 GCUs and then it settles up due to the CPU being saturated. The bandwidth transferred per GCU quickly decreases suggesting to be limited by the server resources. More specifically, most of the resources are occupied by the ControlHub server ensuring UDP reliability via IPbus. In order to meet the 1 kHz trigger rate requirement, the maximum number of GCUs read out per server has to be approximately 100. This result can be further improved, by means of the *jumbo frame* feature implemented within

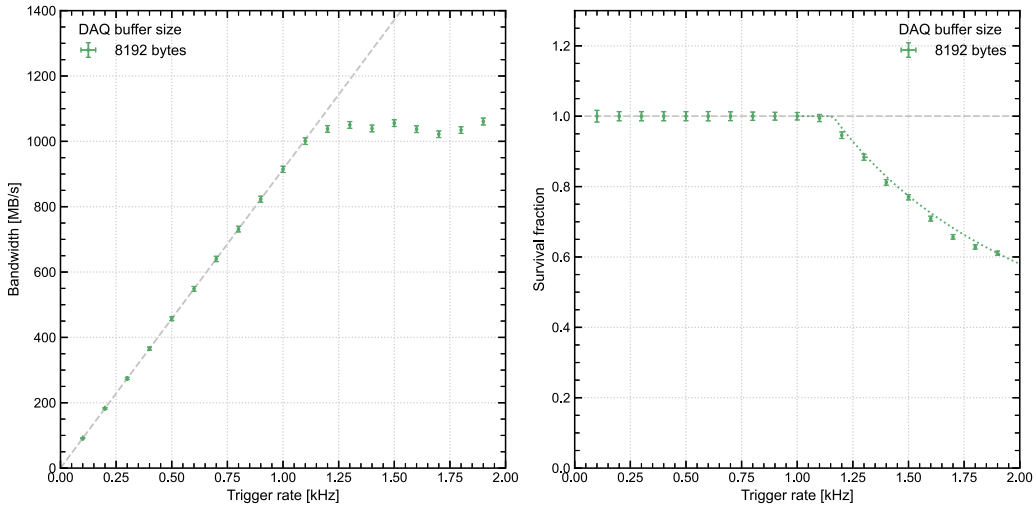

Fig. 10. Transferred bandwidth and corresponding survival fraction for 150 GCUs run in parallel as a function of the trigger rate. Errors are calculated by propagating the experimental uncertainties on the trigger rate and the number of events acquired.

the IPbus framework. Jumbo frames are Ethernet frames with up to 9000 Bytes of payload [22], overrunning the IEEE 802.3 standard of frames with payloads between 46 and 1500 Bytes. Generally, for one client controlling one device, the read or write latency for a single word is approximately 250 μs and the protocol becomes optimal by concatenating multiple transactions into each packet [23]. The network interface implemented on the GCU permits to reach Maximum Transmission Units (MTUs) up to 4000 Bytes, against the standard 1500 Bytes. Increasing the MTU leads to a reduced number of network packets exchanged leading to a lower CPU usage by the ControlHub. The resources occupied by the acquisition itself do not change. Consequently, the relative bandwidth trend is shifted and this allows to reach higher rates without losing events or, equivalently, to employ up to 150 GCUs and still withstand the 1kHz target trigger rate (Fig. 10).

5. Conclusion

The JUNO experiment is expected to inaugurate a new precision era in the field of neutrinos. To reach its aims and the desired energy resolution, the experiment will make use of a total of 20,012 20-inch Large-PMTs. A key requirement for the success of JUNO is therefore the development of a read-out electronics capable of reliably managing, digitizing and acquiring the waveforms coming from the PMTs. This is achieved thanks to the employment of a custom dual-FPGA GCU system installed underwater in the proximity of the PMTs. Following CMS, ATLAS and ALICE successful implementation and testing within their trigger systems, the IPbus protocol has been chosen as the designated transport layer for the readout of JUNO DAQ data and slow control parameters. Two test facilities have been set up in Kunshan and Legnaro to test the performances of the IPbus-based readout of the Large-PMT electronics during the mass production. Carried out measurements showed that a single GCU is able to manage trigger rates up to about 10 kHz without any event loss, thus meeting and exceeding the JUNO design requirement of 1 kHz. When dealing with multiple GCUs run in parallel, the data acquisition performances are limited by the CPU resources of the DAQ server. After careful optimization in the data acquisition and network parameters, it has been possible to manage with a single 24-cores server up to 150 GCUs, still meeting the required 1 kHz trigger rate. A dedicated parallelized software has been developed for an efficient and reprogramming of the GCUs' firmware via Ethernet. Performance tests performed on a 250 set of GCUs in Kunshan show that the developed procedure will permit to completely reprogram the JUNO front-end electronics in about 15 min.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Alberto Garfagnini reports financial support was provided by Government of Italy Ministry of Foreign Affairs and International Cooperation. Xiaoshan Jiang reports financial support was provided by National Natural Science Foundation of China.

Data availability

Data will be made available on request.

Acknowledgments

Part of this work has been supported by the Italian–Chinese collaborative research program jointly funded by the Italian Ministry of Foreign Affairs and International Cooperation (MAECI), Italy and the National Natural Science Foundation of China (NSFC).

References

[1] A. Abusleme, et al., JUNO, JUNO physics and detector, Prog. Part. Nucl. Phys. 123 (2022) 103927, http://dx.doi.org/10.1016/j.ppnp.2021.103927.
[2] A. Abusleme, et al., Sub-percent precision measurement of neutrino oscillation parameters with JUNO, Chinese Phys. C. 46 (12) (2022) 123001, http://dx.doi.org/10.1088/1674-1137/ac8bc9.
[3] A. Abusleme, et al., JUNO, Calibration strategy of the JUNO experiment, J. High Energy Phys. 03 (2021) 004, http://dx.doi.org/10.1007/JHEP03(2021)004.
[4] Y. Han, Dual Calorimetry for High Precision Neutrino Oscillation Measurement at JUNO Experiment (Ph.D. thesis), AstroParticule et Cosmologie, France, Paris U. VII, APC, 2021, URL https://theses.hal.science/tel-03295420.
[5] A. Abusleme, et al., Mass testing and characterization of 20-inch PMTs for JUNO, Eur. Phys. J. C 82 (12) (2022) 1168, http://dx.doi.org/10.1140/epjc/s10052-022-11002-8.
[6] C. Cao, et al., Mass production and characterization of 3-inch PMTs for the JUNO experiment, Nucl. Instrum. Methods A 1005 (2021) 165347, http://dx.doi.org/10.1016/j.nima.2021.165347.
[7] M. Bellato, et al., Embedded readout electronics R&D for the large PMTs in the JUNO experiment, Nucl. Instrum. Methods A 985 (2021) 164600, http://dx.doi.org/10.1016/j.nima.2020.164600.
[8] C. Ghabrous Larrea, et al., Ipbus: a flexible ethernet-based control system for xTCA hardware, J. Instrum. 10 (02) (2015) C02019, http://dx.doi.org/10.1088/1748-0221/10/02/C02019.
[9] F. Marini, Development and Testing of the large PMTs Front-End Electronics for the JUNO Experiment (Ph.D. thesis), Padova University, 2021, URL https://www.research.unipd.it/handle/11577/3443916.

[10] V. Cerrone, et al., Validation and integration tests of the JUNO 20-inch PMTs readout electronics, arxiv, 2022, http://dx.doi.org/10.48550/arXiv.2212.08454.
[11] V. Cerrone, Characterization of the Final Readout Electronics for the Large Pmts of the JUNO Experiment, Padova University, 2021, URL https://thesis.unipd.it/handle/20.500.12608/936.
[12] R. Triozzi, Supernova Neutrino Detection in JUNO, a Large Liquid Scintillator Neutrino Detector, Padova University, 2021, URL https://thesis.unipd.it/handle/20.500.12608/22257.
[13] A. Coppi, et al., Mass testing of the JUNO experiment 20-inch PMTs readout electronics, arxiv, 2023, http://dx.doi.org/10.48550/arXiv.2301.04379.
[14] A. Coppi, Design and Development of Data Quality Monitoring Protocols for the Integration Tests of the JUNO Large PMT Electronics, Padova University, 2021, URL https://thesis.unipd.it/handle/20.500.12608/1067.
[15] R. Frazier, et al., Software and firmware for controlling CMS trigger and readout hardware via gigabit ethernet, Phys. Procedia 37 (2012) 1892–1899, http://dx.doi.org/10.1016/j.phpro.2012.02.516.
[16] G. Anders, et al., Hardware and firmware developments for the upgrade of the atlas level-1 central trigger processor, J. Instrum. 9 (01) (2014) C01035, http://dx.doi.org/10.1088/1748-0221/9/01/c01035.
[17] D. Finogeev, et al., Readout system of the ALICE fast interaction trigger, J. Instrum. 15 (09) (2020) C09005, http://dx.doi.org/10.1088/1748-0221/15/09/c09005.
[18] Advanced Micro Devices, Vivado lab 18.03, 2022, URL https://www.xilinx.com/.
[19] FPGA CORES, FPGAprog utility, 2022, URL https://www.fpga-cores.com/.
[20] B. Jelmini, et al., Characterization of the JUNO large-PMT readout electronics, J. Phys. Conf. Ser. 2156 (1) (2021) 012201, http://dx.doi.org/10.1088/1742-6596/2156/1/012201.
[21] D. Pedretti, et al., Nanoseconds timing system based on IEEE 1588 FPGA implementation, IEEE Trans. Nucl. Sci. 66 (7) (2019) 1151–1158, http://dx.doi.org/10.1109/TNS.2019.2906045.
[22] Ethernet Alliance, Ethernet Jumbo Frames, 2009, URL http://www.ethernetalliance.org/wp-content/uploads/2011/10/EA-Ethernet-Jumbo-Frames-v0-1.pdf. (Accessed 20 October 2021).
[23] T.I.D Group, IPbus performance, 2016, URL https://ipbus.web.cern.ch/doc/user/html/performance.html.