Proceedings of ICRC 2001: 2887 © Copernicus Gesellschaft 2001

ICRC 2001

The ARGO-YBJ Level-1 DAQ System

A. Aloisio [1,2], A. Anastasio[1], F. Barone[1,3], P. Branchini [4], S. Cavaliere [1], V. Masone[1], S. Mastroianni[1], P. Parascandolo[1], and the ARGO-YBJ Coll.[*]

[1]Dip. di Fisica Universitá di Napoli e INFN sez. di Napoli, Napoli, Italy
[2]Universita' del Sannio - Benevento, Italy
[3]Universita' di Salerno, Salerno, Italy
[4]INFN Sezione di Roma III, Roma, Italy
[*]see ARGO-YBJ Coll. list

Abstract. The ARGO-YBJ experiment is presently under construction at the Yangbaijing High Altitude Cosmic Ray Laboratory (4300 m a.s.l), 90 km North to Lhasa (Tibet, People's Republic of China). ARGO will study foundamental issues in cosmic ray and astroparticle physics by detecting small size air showers. The detector covers 74×78 m^2 with single layer Resistive Plate Counters (RPCs), surrounded by a partially instrumented guard ring. The ARGO Level-1 Data Acquisition System is designed around custom protocols and hardware read-out controllers developed to achieve high data transfer rate and event building capability without software overhead.

In this paper we describe the architecture of the system and the hardware developed in the Level 1 environment to acquire the detector's data.

1 The ARGO-YBJ Read-Out system

The building blocks of the ARGO detector (ARGO-YBJ Coll. (1996)) are single-gap RPCs, each with 80 read-out strips 6 cm wide and 62 cm long. Each strip is equipped with a front-end amplifier and discriminator mounted on the chamber edge on a Front-End card. The 8-fold modularity of this card defines a logical partition of the chamber called PAD. The detector is clustered in 6×2-chamber units with modular read-out and trigger electronics housed in Local Stations. In each Cluster, the 960 pertaining strips are sampled with a time resolution of ~ 1 ns by digital, multi-hit TDCs. For each PAD, a fast-or signal is also produced. The entire detector comprises 154 Clusters. PADs are the "timing pixel" of the detector and the trigger logic selects the events on the basis of their multiplicity and distribution in time in the Clusters. The trigger signal acts as a common stop for all the digital multi-hit TDCs reading the detector's strips in the Local Stations. In each Cluster, the Local Station assembles a Data Frame containing an incrementing event number, the addresses of the fired detector's strips and all the timing information retrieved from the TDCs. All the Local Stations are connected with a star-like custom network to a Central Station where the main trigger logic as well as the read-out system are located. Data Frames originated at Cluster-level are pushed into the ARGO Memory Board (AMB) entering the Level-1 read-out system.

ARGO-YBJ adopts a two-layer read-out architecture developed originally for the KLOE experiment (A. Aloisio et al. (1995)). The Level-1 environment is based on crates equipped with VMEbus and AUXbus, a custom high-speed bus that uses the line left undefined by the VME standard. In each crate, a companion read-out controller - the ROCK (A. Aloisio et al. (1996)) - manage the data transfer through the AUXbus.

In order to be scalable, the system is divided in modular structures of VME crates tied by a vertical connection. These chains (Fig. 1) are made of up to 8 crates, each with up to 16 AMBs and the ROCK. The vertical connection links all the ROCK boards to the Level-2 chain controller - the ROCK Manager. Each chain has a unique VME processor board, which resides in the crate of the chain controller. A VICbus interconnection scheme allows the VME CPU to address all the ROCKs and the DAQ boards installed in the chain.

The ROCK performs crate level read-out and gathers data from the AMBs using the AUXbus, a custom protocol developed to enhance the event-driven behaviour of the DAQ system. With a single broadcast transaction (trigger cycle), the AUXbus allows the crate controller to find the AMBs with valid Local Station's frames for a specific event number. Data transfer is then carried out using high-speed random-length block transfers, with an asynchronous VME-like handshake (data cycle). The ROCK then builds data frames consisting of an event number, slaves' data and a parity word.

In the same fashion, the ROCK Manager performs chain level read-out, collecting the data frames belonging to a given event number from all the ROCKs. In order to extend the event-driven behaviour at the chain read-out, the Manager implements an AUXbus' companion protocol, the Cbus. Sim-

Correspondence to: A. Aloisio (aloisio@na.infn.it)

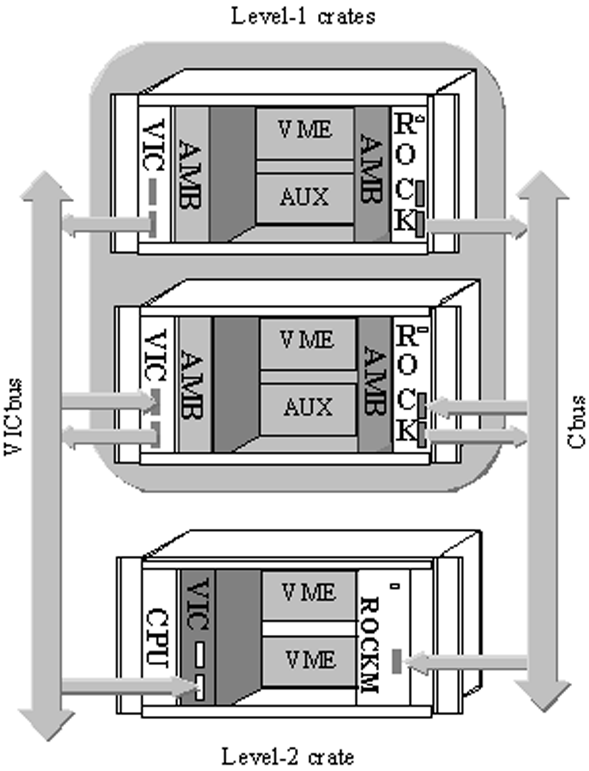

Fig. 1. The Read-out chain.

ilarly to the AUXbus, the Cbus tags data transactions with event numbers, making the entire chain an event-number-driven machine.

Apart from the system initialisation, the data taking process of the chain is fully handled by the ROCKs and the ROCK Manager and does not require any VME activity. For each event number, the ROCK Manager assembles a common data frame containing the respective data from all the ROCKs. In this way, each chain shares also the functionality of a sub-event builder. The VME processor adjacent to the Manager is then in charge of moving the frames on the VMEbus for the next level of processing. In the Level-2 DAQ environment, the handling of the ROCK Manager's frames is the main real-time task of the processor.

The read out process carried out at crate and chain level is fully event-driven and the controllers handle it with no software overhead for the VME processors. When a trigger is issued, data transfers are started by the controllers depending on the presence of an event number to be served.

The introduction of auxiliary busses relaxes the requirements for the VME CPUs and allows using the VME as a channel for monitoring the system performances. On the other hand, in the Level-1 crates all the DAQ board must be designed with both the VME and AUXbus interface. In this framework we designed the AMB, in order to implement all the logic needed to bring to the AUXbus the Local Stations' data frames.

2 The ARGO-YBJ Memory Board

The AMB is a VME double height slave board with A32/D08(O), D16, D32 and D32:BLT data transfer capabilities. The block diagram in Fig. 2 shows the regular and flexible resources' allocation scheme adopted for this board. The input stage is made of four identical slices, each handling a 16-bit data path. Differential data in P(ositive)ECL standard are received and translated in TTL level. PECL logic has been selected for the superior performance in terms of sustainable data rates and and cable driving capability using cheap twisted-pair media. The PECL standard rimoves the hassle for a negative power supply, offering high-density package solutions and a good power to performance ratio. The line receivers output is fed to the SRAM-based Front-End FPGA (XC3164A Xilinx), which redirects the data flow towards an asynchronous dual-port 32Kword FIFO bank.

The input slice has a straight receivers-logic-FIFO architecture in order to implement the widest hardware portfolio throughout custom bitstreams. The FE-FPGAs in the four input slices are daisy-chained, allowing each device to download different configuration data. The FIFO banks decouples the slices'data traffic from the core inner logic of the board - the Back-End FPGA (XC3164A). The BE-FPGA receives the FIFO flags from all the slices and manages the read-out accordingly, gaining full access to all the AMB input channels. Processed data can be directed on the AUXbus to be read out by the local Level-1 controller or on the VMEbus. For each bus, a XC3164A FPGA implements the control logic.

The VMEFPGA contains most of the internal registers to program the board and acts as a gateway to access the internal resources. Through the VMEbus, the most relevant parameters of the input slices and the core logic can be set and read-back accordingly to the bitstream configuration of the companion FPGAs. The FIFO status flags and the AUXbus heartbeath signals are available for inspection in the board memory map.

The AMBs process the Local Stations' data frames in the Level-1 environment. Each input slice receives the data stream originated by one Cluster and parses the data frame syntax. The FE-FPGA stripes off headers and footers and writes in the FIFO bank the data payload, containing event number, addresses of the fired strips in the Cluster and timing informations from the TDCs. The BE-FPGA collects the four event fragments for a given event number, building up a 4-Cluster image of the detector's response. Data can be transferred across the AUXbus as in normal operations or using the VME for stand-alone operations. The AMB also supports a diagnostic mode which allows the user to write test patterns in each input slice, emulating the Local Stations' output. In this way, an extensive check of the entire data flow can be performed, including the frame parsing in the FE-FPGA, the FIFO integrity and the event building in the BE-FPGA. The layout of the AMB is shown in Fig. 3.

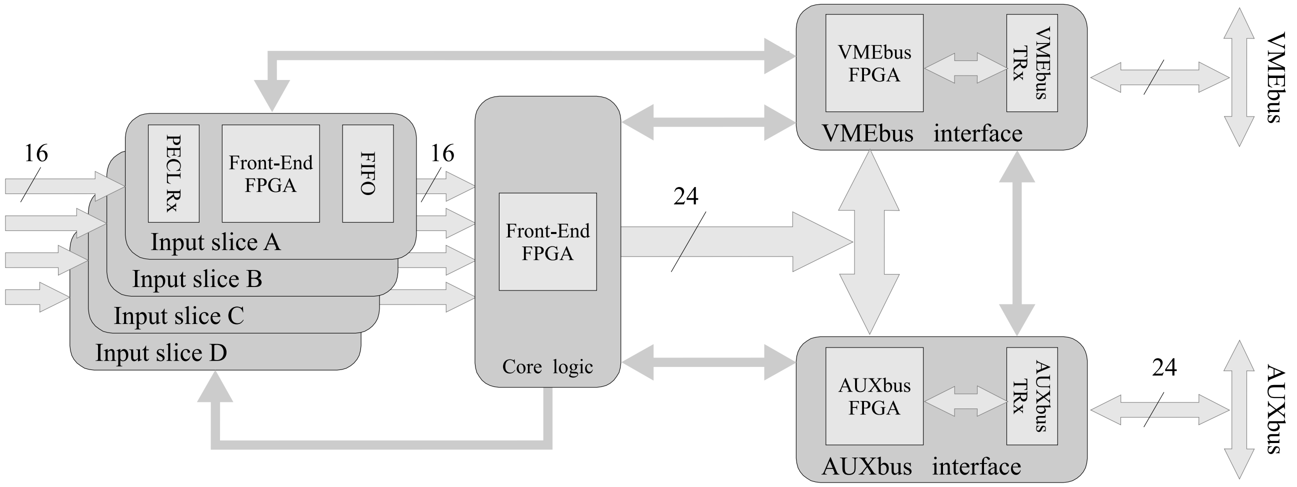

Fig. 2. The AMB block-diagram.

3 The Level-1 Protocol

The AUXbus protocol specifies the rules for transferring data across the bus in the Level-1 DAQ crates. In order to implement a true event driven architecture, AUXbus requires that both the electronics in the Local Stations and the ROCK controller share the same trigger pulses. When a trigger is generated, each sub-system increments its own event number counter. The Local Stations read out the TDCs and the detector strips, label the acquired data with the respective event number, then push the frame into the assigned AMB boards' input slices. In the same fashion, the ROCK stores the event number in a trigger queue and proceeds to readout the AMB boards on an event-by-event basis, retrieving the event number to be processed from the trigger queue.

A fully compelled handshake is used to exchange data between the AMB boards and the AUXbus controller. Some of the key features are broadcast operations labeled by event numbers, block oriented data transactions and a logical and geographical addressing scheme which allows handling up to sixteen slave boards per crate.

The AUXbus protocol has three phases: a sparse data scan, a data readout cycle and a synchronization-check cycle. An AUXbus transaction always starts with a sparse data scan. The event number is broadcast to all the AMBs onto a trigger bus, The boards with valid data belonging to that specific event, assert their individual flag. In this way the data distribution for the processed event number is mapped into a parallel pattern. The AUXbus controller then establishes a connection with the responding AMBs using a 4-bit address bus, which gives the geographical location of the slave board to be accessed. The readout cycle then proceeds by means of random length block transfers. If no AMBs with valid data were found during the sparse data scan phase, the controller skips the readout cycle and starts processing the next event number in the queue with a new broadcast.

AUXbus is fully asynchronous: the data bus is handled by a data strobe - data acknowledgement, VME-like handshake. Different from the VMEbus, slave boards flag their last piece of data transferred using a specific End-of-Block line. This terminates the connection and allows the controller to address the next board in the queue or to end the event number processing if no more boards with data remain.

For the AUXbus system to operate properly, each AMB must contain the same event number in all the 4 headers of the Local Station's data frames. Thus, another feature of the protocol is the synchronization-check cycle. It is performed in the same manner as a readout cycle, except that instead of requesting data, the controller asks for the current value of the event number. The ROCK will then check that all the AMB slices are synchronized.

AUXbus does not support either interrupt handling, or multi-master operations. Data transactions are unidirectional (Read only) and always performed in blocks. Hence, AUXbus is not a replacement for a general purpose protocol like the VMEbus. AUXbus, instead, is targeted to applications such as front-end readout in HEP experiment, where both event driven performances and real time response are crucial.

The protocol has been optimized to achieve high performances in block transfer mode. It overcomes the limitation of a fixed length packet and introduces a sparse data scan labeled with an event number. Used together with the VMEbus, it offers the needed power to squeeze a subevent builder in each front-end DAQ crate.

4 Conclusions

For the ARGO-YBJ experiment, we have developed the AMB board to process the Local Stations'data frame. The AMB implements the AUXbus interface as required by the Level-1 architecture and also supports the VMEbus for stand-alone operations. On the AUXbus, the AMB board features a sustained data transfer rate of 40 MByte/s, allocating a 10 Mbyte/s data bandwidth for each input slice.

In the ARGO read-out system, the event building process

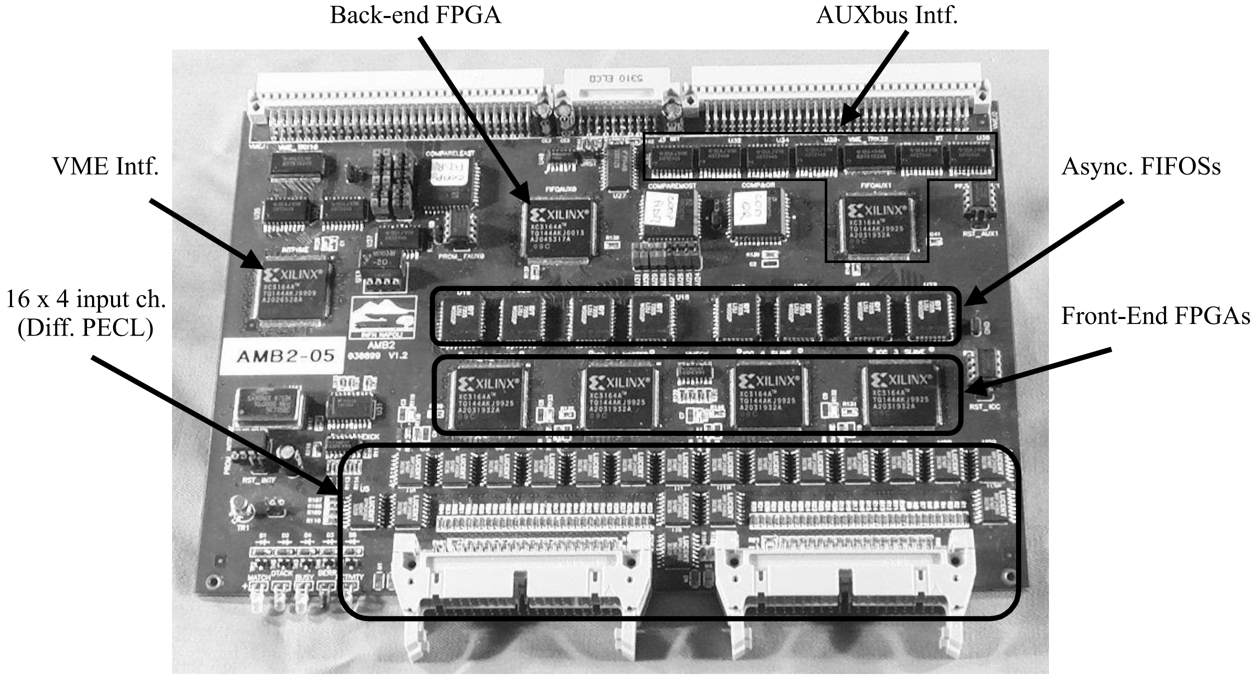

Fig. 3. The AMB layout.

begins just in the Local Stations, where a cluster event is packeted for each trigger. Each AMB board manages four clusters: each slice checks the data syntax and the back-end logic writes a coherent data frame labeled with the event number. At crate level, the ROCK collects from the AMBs the packets related to the event number presently processed, then it builds a new frame and pushes it in an on-board FIFO buffer. The Level-2 controller acts in the same fashion, extending to the chain the event collections.

This read-out architecture permits true event driven operations and pushes the most time consuming operations - such as the subevent assembly and the data framing - to the hardware and to the protocol itself. The event fragments ”agglomerates” each other during the data path in hierarchical structures which eventually contain the entire detector’s response for a given trigger. The time spent in moving data between different nodes includes the negligible overhead required to format the frames. The block-transfer engines implemented in FPGAs manage the data transfer and at the same time merge the transferred packets in the higher-level carrier structures.

The major benefit of this architecture is represented by a complete event reconstruction hidden in the data acquisition process. The Level-2 CPU can then move the ROCK Manager’s data to the on-line computer farm by means of standard network protocols.

This data acquisition system has been successfully tested in the KLOE experiment, showing stable and reliable operations in more than two years of data-taking activity. In the ARGO-YBJ experiment, the Level-1 read-out architecture has been tailored to fit the specific requirements of the detector. The hardware design is now complete and a fully-equipped chain is presently being used to characterize the RPCs performance.

References

ARGO-YBJ Coll., *Astroparticle Physics with ARGO*, Proposal, 1996.
(see also http://www1.na.infn.it/wsubnucl/cosm/argo/argo.html)

A. Aloisio et al., *Level-1 DAQ for the KLOE Experiment*, Proc. of the Int. Conf. on Computing in High Energy Physics (1995) 371, World Scientific Publishing.

A. Aloisio et al., IEEE Trans. on Nucl. Sci. 43, 1 (1996) 167.