CMS CR 2003/036

CMS Conference Report

September 30, 2003

FEDkit: a design reference for CMS data acquisition inputs

V. Brigljevic[a], G. Bruno[a], E. Cano[a,*], S. Cittolin[a], S. Erhan[b], D. Gigi[a], F. Glege[a], R. Gomez-Reino Garrido[a], M. Gulmini[a], J. Gutleber[a], C. Jacobs[a], M. Kozlovszky[a], H. Larsen[a], I. Magrans de Abril[a], F. Meijers[a], E. Meschi[a], S. Murray[a], A. Oh[a], L. Orsini[a], L. Pollet[a], A. Racz[a], D. Samyn[a], P. Scharff-Hansen[a], C. Schwick[a], P. Sphicas[a,c], J. Varela[a]

[a] *CERN, Geneva, Switzerland*
[b] *University of California, Los Angeles, USA*
[c] *University of Athens, Athens, Greece*
[*] *Presented by Eric Cano <Eric.Cano@cern.ch>*

Abstract

CMS has adopted S-LINK64 [1] as the standard interface between the detector front end readout and the central Data Acquisition (DAQ) system. The S-LINK64 is a specification of a FIFO-like interface. This includes mechanical descriptions of connector and daughter board format and electrical signal definition. The hardware/software package described in this paper (FEDkit) emulates the central DAQ side of this interface at the data rate required by the final DAQ system. The performance, integration with the CMS DAQ software framework, and plans for future developments for the DAQ input interface are also presented.

Presented at *LECC 2003*, Amsterdam, Netherland, September 29-October 3, 2003

1 Introduction

The data acquisition system of the CMS experiment has to collect and assemble data from all events accepted by the Level-1 trigger. An event is a set of ~640 fragments originating from an equal number of sub-detectors' Front End Drivers (FED). The nominal Level-1 trigger rate will be up to 100 kHz, and the event fragments are expected to have a maximum mean size of 2kB. All fragments of a given event are delivered to a single Filter Unit (FU), running the High Level Trigger (HLT) to select events for permanent storage. Many FUs will be necessary to reach the required computing power for the HLT. The CMS DAQ architecture [2] is shown in Figure 1.

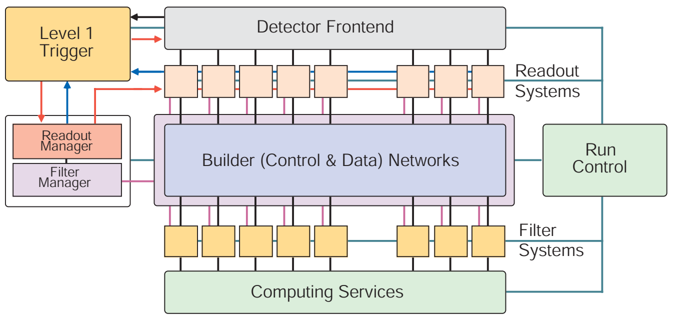

Figure 1: The CMS DAQ architecture

The interface between the FEDs and the DAQ must handle the 100 kHz input rate while sending data at the same speed over the builder networks. Since each sub-detector has its own FED design, the S-LINK64 specification is used as a standardized output interface for the FEDs [3].

On top of the hardware and link-level specification of the S-LINK64, a common data format has been defined. The format encapsulates the sub-detectors' specific raw data in a common envelope. The envelope consists of a header and a trailer that consist of a 64-bit word each. Header and trailer words are flagged as control words in the S-LINK64 stream. The event fragment structure is shown in Figure 2. Notable fields are:

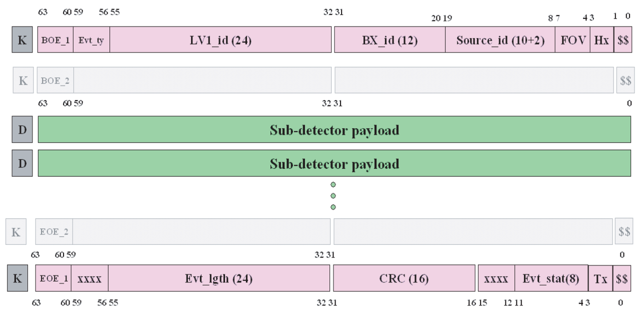

Figure 2: The common data format

Level-1ID (LV1_id), bunch-crossing number (BX_id), identification of the FED for each fragment (Source_id), beginning of event flag (BOE), end of event flag (EOE), and 16-bit CRC protecting the whole fragment, including full header and most significant 32 bits word of trailer (CRC).

The FEDkit consists of a set of boards that allows readout of the S-LINK64 part of the FED from a PC. A daughter board plugged on the FED and another card residing in the PC communicate through a Low Voltage Differential Signaling (LVDS) cable. It is a testing tool for the FED designers.

In the final system, the data from the FEDs will be received by the Frontend Readout Link board (FRL), a 6U compact-PCI board, over a short cable using LVDS. The data are then sent to the surface on optical fibers. By reprogramming FPGAs, the FEDkit hardware boards where used as initial prototypes for the FRL architecture. The FRL design evolved from the FEDkit design.

2 FEDkit hardware

The FEDkit hardware (see Figure 3) consists of the following elements:

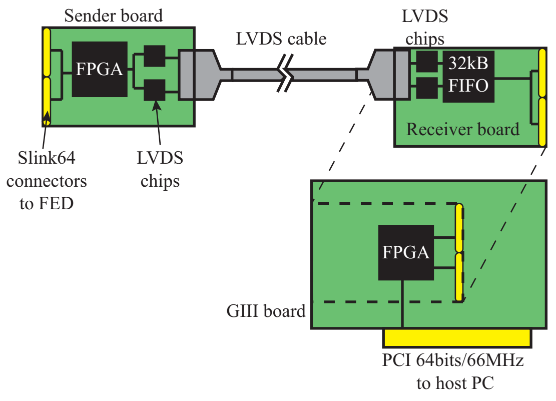

Figure 3: The FEDkit hardware chain

- A sender daughter board that plugs on the FED's S-LINK64 connector. This board generates the link clock. Depending on the length of LVDS cable used, the clock rate can be varied between 60 and 80 MHz.

- A receiver board, also mounted on an S-LINK64 connector. The receiver board can receive and merge the fragments from up to 2 links. It can also be used as a single-link receiver.

- A Generic III board (GIII) [4]. The GIII is a PCI 64-bits, 66 MHz board with an S-LINK64 connector. The GIII, typically used in a PC, is FPGA-based. The PCI 64/66 bus allows a theoretical bandwidth of 528MB/s to the host's memory. A preliminary version of the PCI-X protocol at 66MHz was successfully tested on the GIII but, due to FPGA and PCB limitations, 100MHz clock speeds caused hardware instabilities. The FEDkit is therefore only available as a PCI 2.1 board.

- An LVDS cable. The LVDS cable connects the sender and receiver daughter boards. It contains 14 twisted pairs with a drain wire for each pair, plus a global shield. Various lengths of cable from different manufacturers were tested. The maximum usable length with no errors occurring during tests was 17 m. In this case, the LVDS chips were driven at 60 MHz, allowing a maximum throughput of 480 MHz. The results of the cable test are summarized in Table 1.

- Optionally, the GIII's FPGA can also be programmed to behave as a FED. In this case, the sender board can be plugged on it. This setup allowed the development of the FEDkit components without a FED.

The sender daughter board and the GIII are FPGA-based boards. On the GIII the FPGA contains the PCI interface, the logic to communicate with the driver program, and the logic to control the receiver daughter board's FIFOs. On the sender daughter board, the FPGA controls S-LINK64 signals from the FED and sends data to the LVDS chips. It also contains a 32-words FIFO to allow operation of the LVDS chips and the S-LINK64 interface at different clock speeds.

Table 1: LVDS cable test results

vendor	length [m]	test duration	rate[MB/s] / ν [MHz]	remark
AMP	2	1 month	800 / 100	no error
	7.5	8 hours	800 / 100	no error
	2+7.5+7.5	8 hours	528 / 66	no error (3 cables connected to each other)
Amphenol	15	4.5 hours	528 / 66	no error
	20	minutes	256 / 33	errors
3M	5	19 days	640 / 80	no error
	10	30 days	480 / 60	no error
	15	30 days	320 / 40	no error

The receiver daughter board contains one 32 kB FIFO per LVDS link. The communication between the GIII and the receiver daughter board is not S-LINK64, as the FPGA on the GIII directly controls the FIFOs on the receiver daughter board.

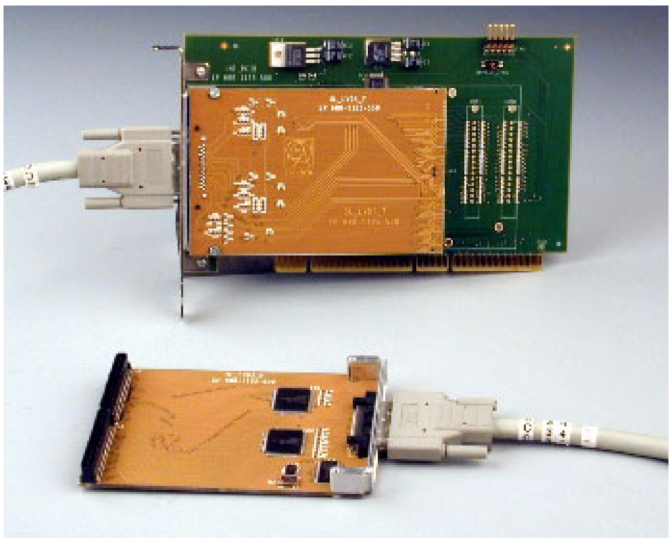
Figure 4: An early prototype of the FEDkit: Generic III with single link receiver (back), sender board (front), and LVDS cable

A picture of the FEDkit boards is shown on Figure 4.

The S-LINK64 interface allows flow-control, and the data stream is stopped in case of congestion. Therefore the FED to DAQ link will not lose data.

3 Software Architecture

The communication protocol connecting the host PC with the FEDkit receiver board was designed with particular attention to performance. The software/hardware protocol relies on FIFOs and pre-allocation of memory blocks, allowing the distribution of free blocks, the DMA of fragment data and the notification of fragment arrival in parallel. The FIFOs are either implemented in hardware, embedded in the GIII FPGA, or in software, using ring lists in the host memory. The blocks are of fixed size, and the hardware transfers the data in one or several blocks via DMA, as required by the fragment size. A schematic representation of the data exchanges between GIII and host PC is show on Figure 5.

Blocks are pre-allocated in host memory by the application and passed to the FEDkit receiver hardware. They are recycled after release by the application.

Incoming fragments are first extracted from the S-LINK64 stream using the control words (headers and trailers). As soon as a header arrives, data are pushed to the memory of the host PC in pre-allocated blocks. When a trailer is received, the DMA stops and the FEDkit receiver writes the size of the fragment received to the word-count FIFO (a ring list in the host's memory). Blocks are used in FIFO order, and it is the task of the host computer program to keep track of the order in which the blocks are passed to deduce which fragment uses which block(s).

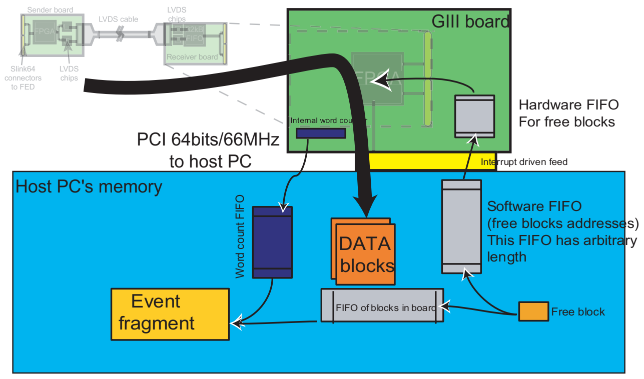

Figure 5: Schematic representation of the data flow between the FEDkit receiver and the host's memory

A special mode of operation allows the dump of the line information (all words transmitted over the S-LINK64, including access to the control bit), bypassing the interpretation of the fragment structure by the hardware. The user can thus see what is received by the FEDkit in case of problems.

The FEDkit software [5] is currently implemented for Linux. It is a set of three layers:

- A device driver
- A user-space library
- Application programs

To optimize performance by avoiding system calls and memory copies, the device driver has been kept to a minimum. Besides basic access to the device (open/close, interrupt handling) all the functions of the hardware are accessed through an OS-bypass (direct mapping of the device in the addressing space of the process).

The user-space library implements high-level functions to receive fragments. The FEDkit can be used with a limited set of functions using defaults parameters. Additional functions can be used to fine-tune the parameters of the FEDkit.

Fragments are stored in block sets in memory. A set of functions can screen the user from the details of the fragment structure at the expense of performance. Direct access to the block structure is also possible.

The FEDkit hardware checks the CRC of the received fragment on the fly and flags fragments with CRC errors. This information is accessible to the programmer using the FEDkit library.

The hardware supports fragment sizes up to 16MB. If a fragment is larger, the situation is handled by the hardware and the fragment is accordingly flagged by the FEDkit library.

Finally, the upper layer (application software or library) can exist in many forms: stand alone programs, or plug-ins for larger frameworks.

A program to test the functionality of the FEDkit has been developed. This program was used to test the LVDS cables as outlined above. A program dumping the link level data of the FEDkit to the terminal has also been written.

The FEDkit has been integrated in the XDAQ [7] framework. XDAQ is a distributed framework used for the CMS DAQ online software. In XDAQ, the FEDkit is integrated as an application. Using this framework, it is easy to create applications that send the fragments over a network (TCP/IP, Myrinet) for event building.

4 FEDkit environment

The FEDkit requires a PC with a 3.3V 64-bits PCI bus. The only supported operating system is Linux. The FEDkit has been tested on PentiumIII motherboards with PCI64/66 buses, Pentium4-Xeon motherboards, and Athlon motherboards.

As stated previously, FEDkit's current goal is to provide a standard test environment for connectivity and performance tests for the FED developers. The FEDkit performance is sufficient to read the FED up to LVDS wire speed (which is over twice the targeted sustained throughput of the FEDs).

The FEDkit has been provided to the FED development groups. The setup FEDkit sender+receiver, with an additional GIII programed as a FED emulator, was successfully tested in test beam with the XDAQ environment. User feedback proved useful in making the FEDkit stable in various PCs.

5 Performance

The performance of the FEDkit was measured for various fragment sizes, on different PCs. The best results were obtained on Xeon PCs. In all cases, the full LVDS wire speed was achieved with sufficiently large fragment sizes.

The results presented in Figure 6 were obtained with a GIII FED emulator as the data source. The LVDS wire speed was 480MB/s. This sustained rate with Pentium Xeon is more than twice the targeted sustained rate of the FEDs (2kB*100 kHz, or 200 MB/s).

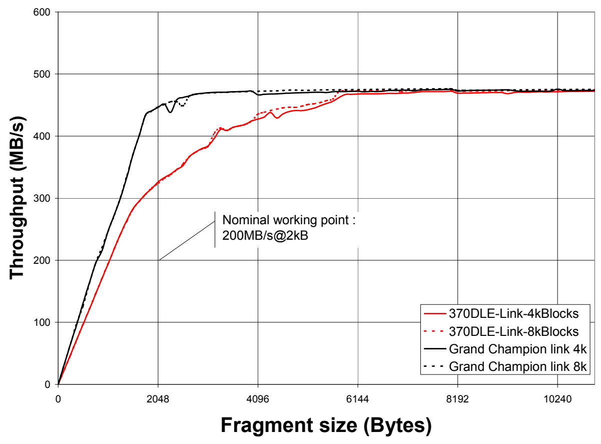

Figure 6: FEDkit throughput against fragment size on PentiumIII (370DLE) and Pentium4 Xeon (Grand Champion) chipsets

The same measurements allow the determination of the average overhead per fragment (FEDkit hardware+FEDkit library) to be 5.3µs on PentiumIII machines and 3.9µs on Xeon machines. This is the average time per fragment for zero payload transferred (just header and trailer). The software part of this overhead is overlapped with the DMA when fragment size is increased. The hardware overhead (which is never masked) is 579 ns per fragment and 182 ns per additional block.

6 FINAL SYSTEM DESIGN: THE FRL

The FEDkit served as a design prototype for the final input of the data acquisition system: the FRL. The FRL is a compact PCI (cPCI) board in 6U form factor. It contains a high-speed PCI-X bus and a bridge to the backplane cPCI bus [6]. The block diagram is shown on Figure 7.

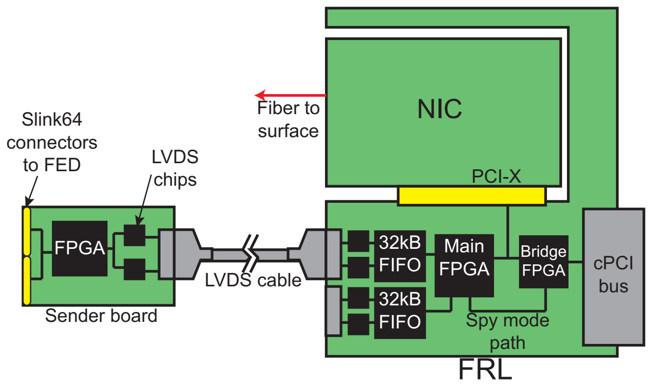

Figure 7: FRL block diagram

The part located on PCI-X contains the S-LINK64 receiver logic, an interface comparable to the FEDkit receiver and a PCI-X slot. The current design hypothesis is to have an intelligent Network Interface Card (NIC) in the PCI-X slot aligned with the PCB plane, on a cutout part of the PCB. The set of FRL plus NIC will fit in a 6U slot of a cPCI enclosure. The FRL board is shown in Figure 8.

The protocol between the FRL and NIC will be comparable to the one in the FEDkit: the FRL will send the data directly to the NIC's internal memory. The receiver side program will run on the intelligent NIC's processor.

The NIC, which is Myrinet in current designs, will send data over a fiber going from the underground counting room to the surface data acquisition building.

The FRL design is based on FPGAs, and this flexibility would also allow a reprogrammed FRL to generate dummy data, allowing DAQ system commissioning without FEDs.

A bridge FPGA, that connects the cPCI bus and the PCI-X bus will also contain a FEDkit-like interface implementing spy function. The spy function will forward a copy of some selected event fragments to the host PC driving the cPCI bus for monitoring purposes. Various selection criteria will be available. It could also be used to acquire data from a given FED, bypassing the downstream DAQ architecture. This could prove useful during detector commissioning.

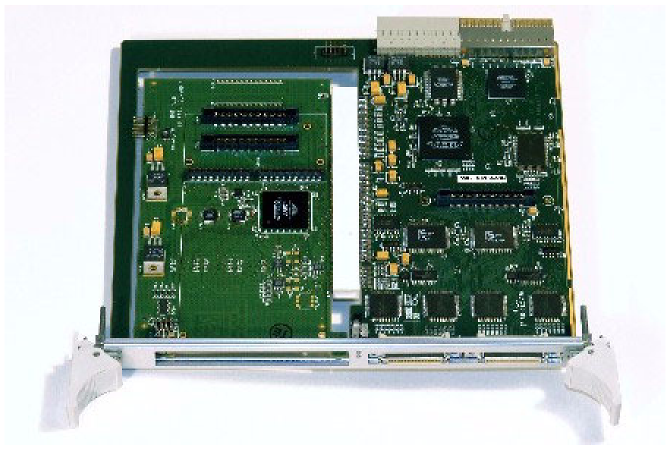

Figure 8: FRL with a GenericIII sitting in the NIC slot (left)

The design of the sender daughter board plugged on the FED remains identical to the FEDkit's. The receiver part of the FRL is also close to the one of the FEDkit. It can receive and merge the fragments from up to 2 links. The LVDS cable is also similar but with a different type of connector, which is mechanically tougher and with better electrical shielding than the ones used on the FEDkits. The cable types will remain the same.

The normal operation at 60MHz or 80MHz will be used for 10m and 5m cables, respectively. In order to cover possible problematic cases where the link clock speed is critical, the senders will be able to drive the LVDS down to 40 MHz. This is a fallback solution as it will not allow the FED to send data at 400 MB/s peak speed as required.

7 Summary

CMS chose the S-LINK64 specification as the uniform interface between all the front-end driver (FEDs) and the central DAQ system. A set of PCI boards collectively referred to as FEDkit, implementing an S-LINK64 interface has been developed.

The FEDkit is based on the Generic III board, and includes a set of mezzanine boards providing an S-LINK64 to PCI interface. The front-end drivers (FED) designers can use this interface as a reference environment to test the integration of the FEDs with the DAQ.

The PCI interface has been developed to minimize interlocking between the software driver and the hardware, hence optimizing the memory usage and speed. On today's (Pentium4 generation) PCs, and at the nominal fragment size (2 kB), the FEDkit achieves around twice the nominal average throughput (200 MB/s) of the FED. Some debugging facilities are added to investigate link-level problems.

The FEDkit can be used stand-alone with the associated library, or inside the data acquisition framework of CMS (XDAQ). Various sub-detector groups are already using it.

During the development of the FEDkit, a sender board had to be developed to provide a data source. This sender board can be used to send pseudo-random data or data from the PC’s memory to its S-LINK64 port, emulating a FED.

The final version of the FEDkit, the FRL, will be used in the CMS DAQ to merge S-LINK64 inputs from up to 2 FEDs and ship the fragment data over a Myrinet link to the event builder.

8 References

[1] S-LINK64 specifications:
http://hsi.web.cern.ch/HSI/s-link/spec/

[2] CMS: The TriDAS Project
Technical design report Volume II:
Data Acquisition & High-Trigger
http://cmsdoc.cern.ch/cms/TDR/DAQ/daq.html

[3] RUWG Readout Unit Working Group
http://cmsdoc.cern.ch/cms/TRIDAS/horizontal/

[4] GIII FED kit
http://cmsdoc.cern.ch/cms/TRIDAS/html/Documents.html

[5] FEDKIT: Interface from S-LINK64 to PCI64/66
http://cmsdoc.cern.ch/cms/TRIDAS/html/Documents.html

[6] Frontend Readout Link board (FRL)
http://cmsdoc.cern.ch/cms/TRIDAS/html/Documents.html

[7] CMS data acquisition (XDAQ)
http://cmsdoc.cern.ch/cms/TRIDAS/html/Documents.html