Design exploration and verification platform, based on high-level modeling and FPGA prototyping, for fast and flexible digital communication in physics experiments

This article has been downloaded from IOPscience. Please scroll down to see the full text article.

2013 JINST 8 P02021

(http://iopscience.iop.org/1748-0221/8/02/P02021)

View the table of contents for this issue, or go to the journal homepage for more

Download details:
IP Address: 131.114.244.130
The article was downloaded on 14/09/2013 at 21:50

Please note that terms and conditions apply.
Design exploration and verification platform, based on high-level modeling and FPGA prototyping, for fast and flexible digital communication in physics experiments

G. Magazzù, G. Borgese, N. Costantino, L. Fanucci, J. Incandela and S. Saponara

ABSTRACT: In many research fields as high energy physics (HEP), astrophysics, nuclear medicine or space engineering with harsh operating conditions, the use of fast and flexible digital communication protocols is becoming more and more important. The possibility to have a smart and tested top-down design flow for the design of a new protocol for control/readout of front-end electronics is very useful. To this aim, and to reduce development time, costs and risks, this paper describes an innovative design/verification flow applied as example case study to a new communication protocol called FF-LYNX. After the description of the main FF-LYNX features, the paper presents: the definition of a parametric SystemC-based Integrated Simulation Environment (ISE) for high-level protocol definition and validation; the set up of figure of merits to drive the design space exploration; the use of ISE for early analysis of the achievable performances when adopting the new communication protocol and its interfaces for a new (or upgraded) physics experiment; the design of VHDL IP cores for the TX and RX protocol interfaces; their implementation on a FPGA-based emulator for functional verification and finally the modification of the FPGA-based emulator for testing the ASIC chipset which implements the rad-tolerant protocol interfaces. For every step, significant results will be shown to underline the usefulness of this design and verification approach that can be applied to any new digital protocol development for smart detectors in physics experiments.

KEYWORDS: Data acquisition concepts; VLSI circuits; Simulation methods and programs; Hardware and accelerator control systems

1Corresponding author.
1 Introduction

In many applications such as High Energy Physics (HEP) experiments, astrophysics, space missions or nuclear medicine it is necessary to develop fast and fault-tolerant data communication protocols and links compatible with harsh working conditions as, for example, high level of radiations. A typical application in HEP experiments is data transfer from detectors (i.e.: silicon pixel or strip sensors) to host units (e.g.: digital data storage and processing units in remote counting rooms). Several types of signals have to be managed by such protocols: raw data plus time, trigger and control signals with different requirements in terms of latency, bandwidth and fault tolerance. A similar data acquisition and communication architecture characterizes also nuclear medicine instrumentation, astrophysics one and space missions. In case of HEP experiments, such as future upgrades of the experiments at the Large Hadron Collider (LHC) [1] at CERN (e.g.: CMS [2], ATLAS [3] and ALICE [4]), dedicated solutions exist for the physical transmission channel such as the GigaBit Transceiver (GBT) [5–7] and the Versatile Optical Link [8]. Such links and the relevant hardware building blocks are specifically tailored to fulfill the requirements of LHC HEP
experiments and mainly implement the physical layer. To define a new communication protocol, flexible and configurable to cover requirements coming from different application fields, implementing also data link and networking layers, and compatible at physical level with several types of transmission links, an integrated design and verification framework is proposed. As it will be shown in the paper the adoption of this design and verification framework led to the definition of an innovative communication protocol, called FF-LYNX, and of IP cores implementing TX and RX interfaces that can be reused and integrated in the front-end of any smart detector.

One of the main issues in the definition of a new communication protocol is the need of reasonable trade-offs between very different requirements: data throughput and data transfer latency on one side, communication hardness and simplicity of interfaces on the other side. In addition, it is important to consider traffic data profiles and typical application errors. Furthermore it is important to develop a parametric tunable system that can be easily adapted to experiments with new requirements. In this paper a top-down design and verification flow optimized for physics applications is described. In the first part, the definition of the communication protocol and the SystemC simulation platform, developed for the evaluation of the technical design options and of the cost metrics, will be treated. Then the VHDL-based design of the interfaces implementing the protocol and of the FPGA emulation system used for their functional validation will be described. In the final part the design of an ASIC test-chip with rad-tolerant prototypes of the interfaces and its FPGA-based test-bed will be treated.

At the state of the art, there is a lack of integrated exploration, design and verification approaches which cover each single step of a complete design flow from the high-level protocol exploration and definition to the test of its ASIC building blocks, integrated with GEANT4 simulations and including also early analysis of the achievable performances when adopting the new communication protocol and its interfaces for a new (or upgraded) physics experiment. The need of such an integrated flow when designing/verifying new systems for HEP experiments, astrophysics or medical physics was expressly underlined in recent works [9]. However, in literature it is possible to find many works, in several application fields, in which just some steps of the whole design flow are analyzed and optimized. Many papers are focused, for example, just on high-level modeling and system verification step such as [10] in which a Simulink-based tool is used. Other works are focused on verification using FPGA-based prototyping platforms [11–15] or mainly on chip testing [16, 17] using SystemVerilog. Similar approaches as that explained in this paper are in [18, 19] where the design and verification step, for consumer electronics devices, is based on Simulink language. Particularly, in [18] a unified algorithm-architecture-circuit co-design environment for dedicated signal processing hardware is presented; in [18] such environment is used for FPGA emulation, ASIC design, verification and chip testing. Similarly [19] presents an environment based on a single design description in Simulink for FPGA emulation, ASIC design, verification and chip testing. These works, in spite of good level of innovation, are not totally integrated. Moreover, the Simulink language adopted in [18] and [19], compared to SystemC language, is more suited for algorithmic design and does not guarantee a good level of granularity in modeling and clock cycle accuracy during simulation, necessary for designing a fast protocol interface. The use of SystemC instead of SystemVerilog such as [16, 17] allows obtaining a high level of flexibility and extensibility of the verification platform even if with lower performance compared to SystemVerilog approach. The platform proposed in this paper, in particular the Integrated Sim-
ulation Environment (ISE) part, thanks to the high-level of abstraction of SystemC, allows a good modeling of protocol interfaces, a good and easy managing of communication between external and internal functional system modules and an highly flexibility to possible modifications, thanks to the Client/Server architecture.

Recently, in [20, 21] the design of the front-end ASIC for the upgrade of the Velopix detector of LHCb experiment at CERN was carried out using transaction level modeling with SystemVerilog and Open Verification Methodology to optimize and verify the architecture before starting RTL-design and synthesis. The same environment was used in [20, 21] also for functional verification of the RTL-implementation. With respect to [20, 21], the approach proposed in this work aims at covering also other aspects of the design and integration of a new components (ASIC or FPGA) within a new or upgraded HEP experiments: at system level allowing also for an early analysis of the achievable performances if using the communication protocol and its interfaces for a new (or upgraded) physics experiment; at component level allowing not only functional verification of the RTL but also automatic generation of the test vectors needed for electrical characterization of the chip before and after irradiation tests.

In this paper an integrated environment covering all exploration, design and verification steps is proposed and applied as example to the new FF-LYNX communication protocol, although it can be applied to any digital protocol development for physics experiments. More in detail section 2 describes the main concepts of the FF-LYNX protocol and the cost metrics used as a reference during its development. Section 3 presents the SystemC simulation environment that guided the definition of the protocol and of the configuration parameters and the evaluation of the expected performances in a given experiment. Section 4 describes the integration with the GEANT4 data generator [22] during the simulation of LHC events, while in section 5 the FPGA based emulation board for protocol model validation is discussed. Section 6 describes the test set-up for the verification and characterization of the ASIC chip that houses the prototypes of the rad-tolerant interfaces implementing the protocol. Conclusions are drawn in section 7.

2 FF-LYNX protocol basis

The FF-LYNX protocol [23] is a “double wire” (i.e.: separate clock and data lines) serial protocol defined at the data-link layer of the ISO/OSI model. It offers a high degree of flexibility as regards data rate and data format. This protocol allows communicating with three different data rates: $4 \times F$, $8 \times F$ and $16 \times F$ (figure 1), $F$ being the frequency of the reference clock. Considering the LHC reference clock (40 MHz) there are 160, 320 and 640 Mbps respectively. The FF-LYNX protocol is characterized by the time multiplexing of two channels, named THS and FRM. The THS channel is used to transmit Triggers, Frame Headers and Synchronization patterns and employs two bits. The FRM channel is used to transmit data packets (information inserted into one or more data frames as figure 2) and employs 2, 6 or 14 bits in the three data rate options. A data packet is a high-level transmission unit that can be formed by several 16 bit words. This data packet can fit a single data frame (if packet is formed by less than 16 words) or can be splitted into several data frames. It uses two kinds of data packets: the Variable Latency Frame (VLF) and the Fixed Latency Frame (FLF) packets, where the latency is defined as the data packet transfer time. The VLF is a generic data frame type while the FLF is used as trigger data frame type. The robustness
Figure 1. Channel partitioning through time-division multiplexing (TDM), with the master clock taken as the period for the TDM cycles.

Figure 2. The basic FF-LYNX frame structure.

of critical information against transmission errors is obtained by means of Hamming codes and custom encoding techniques.

These techniques guarantee the correct recognition of commands and the reconstruction of their timing in the THS channel. In the FRM channel single bit-flips are corrected and burst errors are detected. The use of the same protocol for the transmission of triggers, fixed and variable latency frames brings to a significant reduction of the number of physical links with a positive effect on the overall material budget. All these features make the protocol suitable for the distribution both of DAQ signals and of Timing, Trigger and Control (TTC) signals, that is for the Up-Link (Front-End devices to DAQ system) and for the Down-Link (Trigger and Control System to Front-End devices) paths. On the THS channel, Triggers are higher priority signals with respect to Frame Headers and Synchronization commands; these latter can be transmitted only when there are no Triggers for at least three consecutive clock cycles, in agreement with the current specifications of the LHC experiments. On FRM channel the Data Frames are tagged by Frame Headers transmitted on the THS channel.

Regarding the data format, the structure of data frames is shown in figure 2. The Frame Descriptor (FD) contains information such as the length of the frame and the type of data transmitted;
the Label represents a field that can be employed to add optional information; the Payload constitutes the user data; a Cycle Redundancy Check (CRC) can be optionally applied to the Payload to increase robustness against transmission errors.

As already mentioned there are two categories of data with respect to latency constraints, the VLF and the FLF packets: the former have no latency constraints, while the latter must have a fixed latency. These packets are very important for communication between devices in physics experiments, particularly HEP ones. When Front-End devices detect a particle hit, they acquire and store hit event data. A subset of data is transmitted using FLF packets to the trigger processor that evaluates the Level 1 (L1) trigger. The L1 trigger processor analyzes FLF packet information (e.g.: address of the hit pixels or strips, number of hits, hit timing) generated in a sub-set of the detectors and it generates a L1 trigger only if the event is considered potentially relevant. The L1 trigger is transmitted back to Front-End devices (Down-Link path) on the THS channel to start the download of all the stored hit data associated to the triggered event toward the DAQ system (Up-Link path). Hit data are now encapsulated into VLF packets. The DAQ system finally elaborates all information data for event reconstruction. Hit data to the L1 trigger processor and L1 triggers to the Front-End devices must have a short and constant latency to minimize the storage capability required in Front-End devices and to correctly reconstruct the timing of the triggered events.

The FF-LYNX protocol is implemented in Transmitter (TX) (figure 3(a)) and Receiver (RX) (figure 3(b)) interfaces with a serial port (DAT) on one side and two parallel ports (16-bit port for the VLF packets, 2/6/14-bits port for the FLF ones) with their control (data_valid, get_data, trg) and configuration signals (e.g.: flf_on, label_on) on the other side. Control signals are used by host devices to manage the data transmission operations.

The FF-TX Transmitter is based on the following modules:

- TX Buffer: it is structured as two FIFOs, for storing input data on VLF bus and on FLF bus.
- Frame Builder: it controls the assembly of frames for the transmission of data stored in the FLF and VLF FIFOs.
• THS Scheduler: it works out the arbitration between triggers and frame headers. It receives TRG and HDR commands and passes them to the Serializer.

• Serializer: it generates the serial output stream by receiving the Frame Descriptor field from the Frame Builder and frame words from the VLF/FLF FIFO. It also sends TRG and HDR patterns into the THS channel, according to the THS Scheduler commands.

The FF-RX Receiver is based on the following modules:

• Deserializer: it converts the FF-LYNX serial data stream into parallel form. It separates the THS channel and the FRM channel and provides the data words to store into the RX Buffer.

• THS Detector: it detects the sequences of triggers, headers and synchronization patterns in the THS channel.

• Synchronizer: it generates the reference clock on the base of information coming from the THS Detector.

• Frame Analyzer: it controls the reception of data frames, the data reconstruction and storage in the RX Buffer and the transmission of stored data to the receiver host.

• RX Buffer: it buffers data to be sent to host devices through the parallel port.

3 FF-LYNX protocol modeling

3.1 The ISE (Integrated Simulation Environment) platform

Following an iterative protocol design flow, after the theoretical definition of protocol structure, an ISE (Integrated Simulation Environment) platform, based on models written in SystemC language [24], was developed. An algorithmic model like a Simulink-based one is not suited for our scopes, since the high level model should be much faster than HDL simulation but closely mappable to the HDL model to ensure accurate system-level analysis, architecture design exploration, generation of functional and physical tests vectors, and to be a reliable golden reference for the HDL design step. The SystemC language was chosen to guarantee requirements such as:

• extensibility: it is a measure of the ease with which the model is extensible to add new functionality or to test experimental configurations;

• fast simulations: it is an important requirement if extensive tests are needed;

• granularity: it is intended as the level of abstraction of the language;

• clock cycle accuracy: it is crucial to simulate “real life” behaviors;

The proposed SystemC models describe protocol interfaces, electrical links and I/O test modules with a timing accuracy at clock cycle level. The aim of the ISE platform is to simulate and characterize readout architectures based on the new communication protocol, FF-LYNX in this case study, with input data compatible with possible working environment in which this protocol could
be employed. For this goal, physics GEANT4 input data are used and FoM (Figures of Merit), defined in table 1, are evaluated.

All analysis are conducted in different operating conditions, setting different values of link speed, trigger rate, packet rate, packet average size, bit error rate in electrical serial links. It is possible also to include injection of errors in communication links and memory blocks. All the models that form the ISE are known as “Simulator”.

The developed SystemC link simulator is composed by two main modules: the FF-LYNX TX interface and the FF-LYNX RX interface. This model architecture is parameterized and modular, allowing the reusability of SystemC code and the run-time behavior tuning. This feature is important during the simulation phase when frequent changes in parameter values are needed for FoM estimations.

The Simulator is laid out in a Client/Server architecture (figure 4). In the Server side there are two main blocks, the Test Bench and the Server Main modules, while in the Client side there are the Sim Framework and the Client Main modules. Concerning the Server side, the Test Bench module contains the SystemC protocol interfaces and its task is to transfer input data from the Server Main to the protocol interfaces and then to receive protocol interface outputs. The Server Main behaves as a functional master module; it stores temporarily both data coming from the Client and data waiting to be transmitted back to it.

Both the Server and the Client have a Sim Interface module that interfaces the Server side with the Client side. The message passing is implemented on top of TCP/IP sockets. As regards the Client side, the Sim Framework module is made of a Stim_Gen module that generates the stimuli patterns and a FoM_Gauge module that gauges the figures of merit from the simulation results. The Client Main manages their initialization and provides the highway through which data flows from the Sim Interface module to the Sim Framework module and viceversa.

The ISE architecture, being modular, allows to easily change every single module of the system, as long as the module interface remains the same. Thanks to the Client/Server approach there is a high degree of flexibility since the Server side can be relocated on a different (remote)
Figure 4. The ISE Client-Server architecture; it can be implemented on SMP (symmetric multi processor) workstation or computing grid.

machine or re-implemented for another architecture type (i.e.: FPGA emulator), without modifying the Client side. In this environment a typical simulation is based on one or many “runs” which can have different simulator configurations (parameter settings) to evaluate how the system behaves after these variations. As shown in figure 4, the ISE platform can be also implemented both in Symmetric Multi-Processing (SMP) machines (i.e.: multi-core and/or multi-processor) and in powerful computing grids (i.e. hundreds or thousands of processors) by spreading the load on multiple processing units, in order to decrease the simulation time and conduct longer and deeper analysis.

An example of a simulation carried out in the ISE environment is shown in figure 5 where the packet latency time (mean, max, min and standard deviation metrics), related to VLF data packets varies with the protocol speed (4×, 8×, 16×). For this analysis the physical layer considered is a coaxial cable. In figures 6 and 7 there are two examples of performance analysis that can be carried out, since the early protocol development steps, by using the ISE platform.

The analysis in figures 6 and 7 regards the evaluation of mean sync time and false sync percentage for different values of the N_unlock and of the N_lock thresholds used in the Synchronization module of the FF-RX interface. Sync time, expressed in 40 MHz reference clock cycles, is the time for system re-synchronization after a synchronization loss, while false sync percentage depends on fake synchronization events. N_unlock and N_lock are thresholds that indicate the minimum number of detected synchronization sequences on one of the possible THS channels (4, 8 or 16 in the three speed options) for synchronization unlocking and locking respectively. The synchronization mechanism is based on the counting of THS sequences in each channel and the reaching of two counting thresholds (a high threshold, N_lock, and a low one, N_unlock) was chosen to distinguish
a synchronization lock state (when synchronization is considered as acquired) and an out-of-lock state (when synchronization is being looked for).

This mechanism is hence called Dual Threshold (DT): the transition from the out-of-lock state to the sync lock state takes place when one of the counters reaches the high threshold (thus becoming the in-charge counter) while the inverse transition occurs when a counter that is not in charge reaches the low threshold. The Synchronization module is used for the detection of THS channel and the recovery of the reference clock (“Channel Synchronization”). In these examples three different synchronization algorithms are considered: Privileged Dual Threshold (PDT), Fair Dual Threshold (FDT) and Mixed Dual Threshold (MDT).

Using the PDT algorithm, when a counter hits the high threshold, it resets all the other counters but not itself. With the FDT algorithm, as soon as a counter hits the higher threshold, it resets all

---

**Figure 5.** Mean, minimum, maximum and standard deviation of packet latency with different link speeds.

**Figure 6.** Mean synchronization time for different threshold settings.
the counters including itself. The MDT algorithm combines the two previous variations giving an intermediate level of privilege to the in-charge counter. Taking into account the results of these system-level simulations the PDT synchronization algorithm was used for further development steps of the protocol and hardware implementation of its building blocks. Indeed, with $N_{unlock}$ and $N_{lock}$ equal to respectively 3 and 4, the PDT synchronization technique represents the best trade-off between mean sync time and false sync percentage.

In general terms, the ISE platform allows to tune the parameters of a generic system architecture to design (FF-LYNX protocol interfaces, in this case) according to results of performance analysis and to analyze in detail the behavior of the system during a typical working. In ISE the system architecture to verify is defined using SystemC models. If the system works properly during simulation then, on the basis of SystemC models and of performance analysis, HDL models are defined and implemented onto FPGA. Thanks to the flexibility of ISE platform, it is possible to use the simulator in the FPGA emulator by replacing the server section with the emulator without modifying the client section. As already said, this client-server architecture allows to take advantage of socket communication and to provide a scalable environment suitable for Symmetric Multi-Processing (SMP) workstations or computing grids.

### 3.2 VHDL Test Bench

A VHDL Test Bench, based on the control and readout system of the CMS (Compact Muon Solenoid) pixel detector, was developed for the functional verification and the performance evaluation of the VHDL models of the FF-LYNX interfaces defined on the basis of SystemC models. As it is shown in figure 8 there are three main blocks: a Hit Generator (HG), a Readout Data Collector (RDC) and the Device-Under-Test (DUT). The HG generates the stimuli vectors from a Hit file that contains several information about address and signal amplitude of the hit pixels on the base of real physics experimental data (from LHC). The RDC receives and stores in a Readout file hit

![Figure 7. False synchronization percentage time for different threshold settings.](image)
data from the DUT. The DUT is a control and readout system made of many pixel Read Out Chip (ROC) devices that use FF-LYNX interfaces (red and blue blocks in figure 8) for communication with the Token Bit Manager (TBM) circuit that controls the readout of the ROCs in the CMS pixel detector module. After the end of test run a cross-check of the Hit file and the Readout file is done in order to evaluate the FoMs.

4 ISE applications

The developed ISE environment can be also used for an early evaluation of the impact of a new communication protocol and its hardware implementation on the performance of data acquisition systems; this drives to great advantages in terms of reduction of development time and costs for a new project/experiment or an upgrade of an ongoing one. Particularly, the ISE environment was used to characterize the Track-Trigger architecture proposed for the upgrade of the CMS Silicon Tracker, whose data are used in CMS to reconstruct the trajectories of the charged particles. Achievable data-rate was evaluated comparing different algorithms used for clustering, pairing and track-let finding. The performance of the readout system (e.g.: lost hit rate, corrupted hit rate, cluster rate, pair rate) was analyzed using different protocol configurations (e.g.: size of FLF packets) and hardware implementations (e.g.: Front-End buffer size, number of links, link speed). In performance analysis physics event classes (i.e.: loss rate, corrupted hit rate, cluster rate, pair rate for given physics events) were considered.

By using GEANT4 data as ISE inputs an analysis was conducted; examples of the achievable results are reported in figures 9 and 10 if using the FF-LYNX protocol described in section 2 for the upgrade of the CMS tracker. Particularly, figure 9 shows the achievable data rate, Mbps, at different Z values (longitudinal positions in the CMS Tracker) (Z0 = 0 cm, Z1 = 69 cm, Z2 = 137 cm, Z3 = 206 cm) and for different detector layers while figure 10 reports the achievable transmission efficiency as a function of the effective available bandwidth.
In order to test the FF-LYNX protocol interfaces and to validate the ISE simulation results, an FPGA-based emulator platform was designed. This emulator system is based on a C++ Graphical User Interface (GUI), used for the configuration and the control of the emulator, and on the VHDL emulator core. The VHDL emulator is implemented on an Altera Stratix II GX FPGA device (EP2SGX130GF1508) [25] housed on a commercial PCIe board (PLDA XpressGXII board) [26] that can handle a 3.6 GB/s data rate (effective, full duplex). This FPGA development board was chosen to guarantee a large bandwidth to perform extensive emulations with high data rates with an acceptable efficiency in terms of emulation time. A typical real working scenario is to receive data from a CMS Tracker Front-End chip with a 1.8 Gbps data rate (evaluated from physics simulation). Three or four FF-LYNX TX/RX interfaces working at 640 Mbps (maximum data rate allowed by
2013 JINST 8 P02021

Figure 11. VHDL emulator system.

![Figure 11](image1.png)

Figure 12. FF-Emulator core block diagram.

![Figure 12](image2.png)

the FF-LYNX protocol with a reference clock frequency of 40 MHz) could handle this data rate. The PCIe board, with its large bandwidth, allows realistic emulations. In our case, for one second of emulation, sending the test vectors and receiving the emulation results take about 62.5 ms. In addition, having a single FPGA development board that can be mounted on a workstation, instead of having a platform composed by many separated parts, allows to increase the compactness of the whole emulation system. This is a key feature towards the reuse of the same emulator environment as test bed when doing irradiation tests on the ASIC chips implementing rad-tolerant FF-LYNX building blocks (see further details in paragraph 6.3).

5.2 VHDL emulator system

The VHDL emulator system is formed by three main functional blocks, as shown in figure 11: the PCIe XpressLite core, the Interface Logic and the FF-Emulator core.

The PCIe XpressLite core in figure 11 is an IP-core block provided by PLDA. It manages the communication with the PCIe bus and therefore with the host workstation. The Interface Logic is another block provided by PLDA, but it can be customized by the user and it is used to interface the PCIe XpressLite core to the user custom application block that is the FF-Emulator core. The FF-Emulator is a custom developed core whose block diagram is shown in figure 12.

The FF-emulator core is divided in two modules: a Test Controller (TC) that manages the emulation test and the FF-LYNX TX and RX interfaces that represent the Device Under Test (DUT).
The TC has a simple structure based on the following building blocks:

1. TC core that includes two functional paths. The TX path manages the data transmission toward the FF-TX block. It is made of the TX Controller and the TX RAMs. The RX path manages the data reception from the FF-RX block. It is made of the RX Controller and the RX RAMs;

2. TC configuration registers that contain test parameters as emulation window duration, number of VLF packets, presence or not of FLF packets, size of the FLF packets, TX TC buffer size, etc;

The TX controller starts the transmission of a trigger (or of a FLF packets, when they are enabled) or the transmission of a VLF packet at pre-defined clock cycles (TX Time Stamps). The arrival timing of the received triggers and frames is stored and then compared with the TX to evaluate the trigger and the packet latency. The TX and RX RAMs are divided in five different types of RAM:

- **VLF_Ts** (Variable Latency Frame Time Stamp) RAM to store the time stamps (TS) associated to VLF data packets transmitted/received to/from FF-LYNX interfaces;
- **VLF_LEN** (Variable Latency Frame Length) RAM to store the length (number of 16 bit words) of the VLF data packets transmitted/received to/from FF-LYNX interfaces;
- **VLF_LARGE** (Variable Latency Frame Data Word) RAM to store the VLF data packets transmitted/received to/from FF-LYNX interfaces;
- **TRG_Ts** (Trigger Time Stamp) RAM to store the time stamps associated to triggers or FLF data packets (if they are enabled) transmitted/received to/from FF-LYNX interfaces;
- **FLF_LARGE** (Fixed Latency Frame Data Word) RAM to store the FLF data packets (if they are enabled) transmitted/received to/from FF-LYNX interfaces.

The TX and RX Controllers are made up of (i) time stamp counters whose current values are compared with values stored in the TX TS RAMs or stored in the RX TS RAMs, (ii) FSMs (Finite State Machine) to control the test flow and (iii) two FIFO buffers (only for TX Controller), one configurable buffer to store VLF data temporarily and another to store VLF data lengths associated to VLF data buffered on the first buffer. These two buffers allow loading a complete packet before sending it on FF-LYNX link.

### 5.3 The emulator GUI software

The C++ GUI software is organized as shown in figure 13. It is based on three main parts: the PLDA drivers to interface the Host-PC with the PCIe PLDA board, the Data Manager to manage the emulator operations and the GUI to set all the emulation variables and to display emulation results. In the Data Manager block three important parts can be found: the EmuRun part which manages an emulation run on the base of several parameters set by user (as FF-LYNX interface data rate, emulation window duration, packet and trigger rate, number of triggers and packets, etc); the StimGen part that generates the VLF/FLF data packets and the time stamps and stores them
in the TX RAMs; the figures of Merit (FoM) Gauge that, according to emulation results, allows to define some FoM (as shown in table 1), useful for validating and verifying FF-LYNX protocol efficiency.

5.4 Emulation test

Using the proposed emulation environment it is possible to extract some interesting graphs about the obtained FoM, thus assessing the performances of the communication protocol and its hardware building blocks, and evaluating their suitability when applied to the physics experiment of interest.

As example, next figures show some tests carried out using $8 \times$ FF-LYNX interfaces and links. Particularly, figure 14 shows the achieved packet latency, in 40 MHz reference clock cycles, as a function of the packet size expressed in data words. The achieved results provide important suggestions about the most suited way to configure the communication protocol and its interfaces for the project of interest (e.g. an HEP experiment); indeed figure 14 shows that when the packet size increases over 52 data words there is a sharp increase of the packet latency. This increase is due to packet queuing in TX buffer of FF-LYNX interface under test. Below 52 data words there is a proportional increasing, with an “heuristic law” of roughly 4 clock cycles of latency increase for each new data word in the packet size, simply due to the necessary time to send larger and larger packets.

Figure 15 shows the Packet latency, in 40 MHz reference clock cycles, as a function of the trigger gap, i.e. the distance, still expressed in clock cycles, between consecutive triggers. In these experiments the packet size is changed randomly. Figure 15 shows that when the trigger rate
increases the packet latency increases too. Packet latency increases because trigger priority is higher than data priority and therefore, when increasing the trigger rate, the VLF packets are forced to wait into buffer before being sent.

To be noted that in figures 14 and 15 for the packet latency all 3 main cost metrics defined in table 1 were measured by emulation: min, max and mean PL.

Figure 16 shows the loss rate of data words as a function of the TC buffer size. The achieved results prove that by increasing the TC buffer size it is possible to decrease data word loss, because
of a more data buffering. In figure 17 there is an important result that underlines the importance of emulator approach as a necessary part of protocol interface development and validation. It shows the differences between high level simulation and emulation when measuring, as example, the packet latency as a function of the distance (in 40 MHz reference clock cycles) of consecutive packet transmissions. As reported in figure 17, using a packet size equal to 8 words (16 bit), when

![Figure 16. TC buffer size vs. data word loss rate.](image1)

![Figure 17. Packet latency vs. packet gap.](image2)
packet gap decreases at 30 clock cycles or lower there is a mismatch between simulation results obtained thanks to ISE platform and emulation results obtained with FPGA emulator. The difference is due to protocol modeling inaccuracies proper of the high-level abstraction approach. In the specific case of the FF-LYNX simulation environment the SystemC model of the protocol interfaces does not fit perfectly the real physical protocol interfaces: the model does not consider the finiteness of TX buffer and a consequent possible buffer overflow with packet loss. These packets that would be really lost, instead, are seen as packets still buffered and so there is a fake packet latency increasing in figure 17 for the high level simulation vs. the real-world emulation.

5.5 Protocol test time comparison

A key issue of a design and verification environment is the time needed to simulate or emulate a given time frame of the real system. Hence, several tests were done to assess the protocol test time of the proposed FPGA emulator compared to the protocol test time achievable with other high performance (HP) computing machines or with a distributed computing GRID. The following protocol test conditions were adopted: test time 60 s; speed 4× or 8×; trigger rate 133 kHz or 400 kHz; packet rate 133 kHz or 400 kHz; packet size 5 or 8 words (16 bit).

The HP machines used for the comparison are:

- Intel Core Duo 2 T9300 (45 nm CMOS technology) processor with a 2.5 GHz clock speed, 2 cores, 6 MB of L2 cache and a 64 bit instruction set;
- Intel Xeon X7550 (45 nm CMOS technology) processor with a 2.0 GHz clock speed, 8 cores, 18 MB of L3 cache and 64 bit instruction set;
- INFN (Istituto Nazionale di Fisica Nucleare) PISA GRID data center [27], formed by a network of 400 multicore processors with a total elaboration power of 40 Terafloat/s and a data storage capacity of 500 TeraBytes.

In table 2 the achieved test timing results are summarized. It is worth noting that by using the FPGA emulator platform very fast tests can be performed: 1 minute of real system time can be emulated in roughly 6 minutes and so very long experiments in great details can be carried out. Thanks to our platform, the test time is decreased by a factor of about 20× and 100× compared to HP computing machines such as the 8-core Intel Xeon and the 2-core Intel CoreDuo respectively. The testing time performance of the proposed emulator platform is comparable to the results achievable with a much complex and costly 400 CPU GRID computing center [27].

<table>
<thead>
<tr>
<th>Test Environment</th>
<th>Test Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation on Intel Core Duo@2.5 GHz</td>
<td>11h 47m</td>
</tr>
<tr>
<td>Simulation on Intel Xeon 8-Core@2.0 GHz</td>
<td>2h 21m</td>
</tr>
<tr>
<td>Simulation on INFN GRID (400 CPUs)</td>
<td>3m 21s</td>
</tr>
<tr>
<td>FPGA Emulator</td>
<td>6m 40s</td>
</tr>
</tbody>
</table>
6 “Test Bed” emulator for ASIC chipset testing

6.1 Test bed architecture

After the emulation phase that allowed the functional validation and the characterization of the FF-LYNX interfaces, an ASIC chip implementing FF-LYNX interfaces (called FF-TC1) was designed. The ASIC was produced using the IBM 130 nm CMOS8RF process guaranteeing intrinsic radiation hardness against Total Ionizing Dose (TID) effects. Many radiation hardness techniques against Single Event Effects (SEE) were implemented on the ASIC test-chip such as: triple modular redundancy for registers and Pseudo-Random Generators (PRGs) design; hamming encoding and scrubbing in FIFOs design; interleaving of FIFO cells, placing them manually, following a pattern that spreads bits of the same words reducing the risks of multi-bit faults. The details of the ASIC-design and irradiation test characterization are reported in [13]. Its size is $2 \times 2 \text{mm}^2$ and its total power consumption is less than 40 mW with an input clock frequency of 40 MHz; its package is an LPCC-68 pins. The test-chip includes: the three separate rad-tolerant FF-LYNX TX/RX interfaces ($4 \times$, $8 \times$, $16 \times$), a test controller with PRGs for “built-in” test and I2C interface for control and monitoring, as shown in figure 18.

To perform functional tests and irradiation tests of the ASIC chipset, the emulation system introduced previously allowed the easy and rapid generation of the ASIC test-bed system. The few modifications carried out on the emulator proposed in section 3 to generate the ASIC test-bed...
systems were the following: at VHDL system side, the implementation of I2C module, PRGs and signal path multiplexing; at software system side, the I2C managing and the chip test procedure setting. To configure the test chip, the I2C protocol is chosen decreasing the number of used chip pins. An I/O parallel bus is defined which is used as input or output port when running the tests. Built-in test features, as the PRG integrated in the test-controller on the FPGA and also into the test-chip, are implemented to generate data, packet lengths and time stamps from the test-bed itself. By following this strategy, equivalent PRGs running at the same clock frequency and loaded with the same seeds, generate equal pseudo-random data on the two test-controllers, on the FPGA and the ASIC. That allows a high reduction of connections between the test-bed and the test-chip, and consequently a reduction of pads on the die. In figure 19 it is shown the structure of test connections between the FF-TC1 chip and the Test Bed emulator; red and violet lines are control signals, blue lines are data signals, green lines are serial FF-LYNX signals and orange lines are internal signals. The chip-test procedure is set from the GUI software during the chip and test-bed configuration phase.

Depending on the device under test (DUT) there are 3 main ASIC test configurations supported:

- TX interface on chip: data, lengths and time stamps are generated either on test-bed emulator using its PRGs, controlled by its TX controller (TXC), or on test-chip using its PRGs. In the first case data are sent to back-end of FF-LYNX TX chip interface by means of parallel bus
(configured as input bus); in both cases these information are pre-stored on TX RAMs inside test-bed emulator. TX chip interface sends data to FF-LYNX RX test-bed interface using FF-LYNX serial line and then these data are stored on RX RAMs.

- RX interface on chip: data, lengths and time stamps are generated on test-bed emulator using its PRGs controlled by its TXC and also stored on TX RAMs. Next, data are sent to FF-LYNX TX test-bed interface that transmits data to FF-LYNX RX chip interface using FF-LYNX serial line. In the end FF-LYNX RX chip interface sends data to Test-bed emulator using parallel bus, then these data are stored on RX RAMs.

- TX and RX interfaces on chip: data, lengths and time stamps are generated on test-chip using its PRGs controlled by TXC on chip. At the same time on test-bed emulator, using PRGs with the same seed, these information are generated and pre-stored on TX RAMs. FF-LYNX TX chip interface sends data to FF-LYNX RX chip interface, using FF-LYNX serial line, and then this latter sends data to Test-bed emulator using parallel bus (configured as output bus) for storing data on RX RAMs.

The test finishes when information are loaded from TX and RX RAMs of the Test-Bed Emulator and sent to the PC-host by means of PCIe bus for generating the FoM.

### 6.2 ASIC chip functional verification results

To assess the correct functional behavior of the ASIC chip implementing the FF-LYNX interfaces, several tests were carried out using the test bed system described in section 6.1; moreover all the FoM defined in table 1 were measured. As an example of the achieved results, figure 20 shows the variation of the packet latency, in 40 MHz reference clock cycles, as a function of the packet gap (i.e. the idle clock cycles between consecutive packet transmission), considering a packet size of 6 data words and $4 \times F$ ASIC FF-LYNX interfaces.
As it can be noted from figure 20, when packet gap is lower than 10 clock cycles the TX buffer overflows with packet loss consequently, with a packet gap between 10 and 15 clock cycles there is a packet queuing into TX buffer while after a packet gap of 15 clock cycles there is a sharp decrease of the packet latency due to a decreasing in data buffering. A statistical analysis of the packet latency for the \(4 \times F\) ASIC FF-LYNX interfaces, considering a packet size of 4 data words, on a sample of 150 test runs is shown in figure 21, where a Lorentzian fitting curve is drawn for three fixed values of packet gap (48, 52 and 56 clock cycles). In figure 21 both the mean packet latency, expressed in 40 MHz reference clock cycles, value and its variance decrease when the packet gap increases, in agreement with the actual working of the interface.

6.3 ASIC chip total dose hardness verification results

This Test Bed architecture, as already explained, was used also for irradiation tests in order to verify the Total Ionizing Dose (TID) hardness of the ASIC which integrates the TX and RX FF-LYNX interfaces. As irradiation source the X-rays at 19 keV is chosen. These tests were conducted at CERN in Geneva using a SEIFERT RP149 X-rays generator characterized by a molybdenum anodic target, a 40 kV power supply voltage and a tube current from 6 to 40 mA. The X-ray generator was calibrated on the base of the calibration graphs [28] provided by CERN X-ray facility staff which has allowed to define the dose rate by setting tube current, power supply voltage and distance between collimator and device under test.

The test-bed system was positioned outside the irradiation chamber apart the Test board with the ASIC under test that was placed under the beam at a distance of 1 cm. The irradiation procedure was defined on the base of the method in [29], consisting of five Total Dose steps to reach,
Figure 22. Overall current absorbed with respect to TID during the irradiation phase (1.5 V core supply voltage, 2.5 V I/O supply voltage).

chosen with logarithmic increment: 1° step: 500 krad (SiO$_2$); 2° step: 1 Mrad (SiO$_2$); 3° step: 5 Mrad (SiO$_2$); 4° step: 10 Mrad (SiO$_2$); 5° step: 40 Mrad (SiO$_2$). The dose rate chosen was about 30 krad/min. After every step a room temperature annealing phase followed the irradiation. To be noted that the target of this work is having a flexible communication protocol with relevant IP macrocells for the TX and RX interfaces, that can be integrated by the designers of the detector front-end in several application fields. This is why different radiation levels were considered for the X-ray test.

The FF-TC1 ASIC chip was configured in standby mode during the irradiation time. For all test time (during and after irradiation) the current consumed by the chip was monitored and only after the end of a single irradiation step an automatic loop test was started from remote GUI. This automatic test consists of executing all the three test modes detailed in section 6.1 cyclically within a settable time window of max 30 min while logging results on text file. In the course of these runs, no trigger or packet error came out, proving therefore the robustness of the test-chip against TID effects at least up to 40 Mrad (SiO$_2$). The current consumption level (as seen in figure 22) after the last annealing phase is similar to the one measured before the irradiation, a sign that no destructive events occurred after a Total Dose of about 40 Mrad (SiO$_2$). Figure 22 shows the variation of total current consumption with TID during irradiation phase; in every step there is a current consumption peak whose value decreases when TID step increases.
7 Conclusions

A top-down innovative design and verification flow was developed. In order to demonstrate the validity and quality of this design flow the design of a new communication protocol called FF-LYNX was chosen as a typical case study. This protocol is fast and flexible and is utilisable for several applications in HEP, medical and space research fields. It is candidate to be used as fast data protocol in the 4DMPET project [30], an INFN project about the design on innovative block detector for Positron Emission Tomography (PET), and as auxiliary protocol for the upgrade of the ME1/1 station of the End-Cap Muon (EMU) detector [31] of the CMS experiment at the LHC, that is foreseen during the LHC technical shutdown in 2014. In both the above cases low-dose radiation rates and negligible single-event effects are expected so that the design and characterized ASICs are already suited for adoption. The proposed design approach keeps under control all system variables and parameters thanks to the ISE platform (based on SystemC models) and FPGA-based emulator. The functional, electrical and irradiation tests successfully carried out on the FF-LYNX ASIC confirm the suitability of the proposed approach for an easy generation of the test-bed starting from the already developed ISE (Integrated Simulation Environment) and emulation platforms used in early design steps for: high-level protocol definition, golden reference for HDL generation, early analysis of the achievable performances when using the communication protocol and its interfaces for a new (or upgraded) physics experiment, IP macrocell verification (by simulation and by FPGA emulation), and finally ASIC verification. At all levels of the design flow, from protocol specification down to ASIC realization, proper FoM are evaluated. Using this new approach, project failure risk decreases and also the project development and verification time is reduced. Its usefulness is evident considering the hardware designing, as in the case of FF-LYNX protocol interfaces implemented on rad-hard ASIC chip.

Acknowledgments

This work was supported by INFN group V and by University of California at Santa Barbara; discussions with Dr. N. Randazzo and Dr. A. Aloiso, INFN group V project reviewers, with the GBT group at CERN, with Dr. C. Tongiani and Dr. G. Bianchi are gratefully acknowledged.

References


