Hardware Implementation of a FPGA-based Universal Link for LVDS communications

Luis Sanchez*, *Student Member, IEEE*, Giancarlo Patiño*, *Student Member, IEEE*,
Victor Murray*†, *Senior Member, IEEE*, and James Lyke‡, *Senior Member, IEEE*
*Department of Electrical Engineering, Universidad de Ingenieria y Tecnologia, Lima, Peru.
†Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, New Mexico, USA.
‡U.S. Air Force Research Laboratory, Albuquerque, New Mexico, USA.

***Abstract*—We present the first hardware implementation for a FPGA-based universal link for the transmission of different low-voltage differential signaling (LVDS) connections through a single LVDS connection between different devices. The main objective of this work is to reduce the number of wires in a network, for example in some satellites, with several groups of devices, to a single LVDS connection. This paper proposes a new communication protocol for successfully coding and decoding the data sent through the single connection. We propose a solution for one of the difficulties of LVDS standard due to the amount of wires needed for a duplex connection, significantly reducing the amount of wires required for a large network. The proposed solution has been implemented in an Atlys board with a Spartan 6 FPGA showing promising results.**

***Index Terms*—LVDS, FPGA, universal link**

I. Introduction

Low voltage differential signaling (LVDS) is an electrical signaling standard commonly used in industrial and consumer electronics applications requiring high-speed communications. The simplicity of the process allows a robust and quick determination of the binary value based on the voltage comparison between a pair of wires. Some advantages of LVDS connections are the extremely low power consumption and transmission rates up to 3.5 Gb/s [1].

The simplest configuration for a LVDS physical implementation requires at least 2 wires for a unidirectional communication. More complex configuration requires up to 4 wires for a full duplex communication between 2 devices. This high demand on wires increases very quickly the complexity of the network.

The LVDS physical layer protocol is already used in many areas of industry. However, the focus of this research is the aerospace industry, like Spacewire [2], [3]. The intrinsic advantages of the LVDS protocol offer protection from radiation and other noises that usually affect spacecraft network performance. At onboard networks, the weight and the volume occupied by the devices play an important roll at selection of the components of the network. Because of that, a device capable of connecting all those devices within a minimum of connections offers a great alternative for onboard communications.

We present a solution called 'Universal Link,' capable of fully communicate two groups of devices through a single LVDS connection without any extra configuration in the devices in the network. It is based on a tag-scheme where signals are processed bit by bit, which enables the use of any encryption protocol or another communication protocol like I2C (inter-integrated circuit) (see examples in [4], [5]). Thus, an universal link can provide to all devices with a virtualized network with completely scalability ports able to form the tag-scheme from a generic chain depending on number of inputs, transmit through the LVDS single connection and receive all data with a major throughput (see [6] for an example).

The Universal Link is divided in two devices connected together by a single LVDS connection (see Fig. 1), the multiplexer device, coding the input signals in data packets for serializing along with a tag protocol specially designed for a correct decodification at the demultiplexer device. This second device finds the identifiers and saves the data so it can be serialized again at the original frequency of the signal.

In this manuscript, we propose an improvement over the previous identifier protocol [7]. The new tag protocol guarantees that no data pack could be taken as an identifier. The main objective in this phase of the research includes the hardware implementation of each part of the system in two Atlys boards connected with a LVDS protocol. This board natively supports LVDS within the I/O ports for a quick implementation using the provider modules and tools. Thus, we think that this proposed solution represents a first approach to reduce the number of wires in systems with several LVDS communications.

We present the methodology in section II. Then, in section III, we present the results and discussion for the tests performed. Finally, in section IV, we present the conclusions given this first approach.

II. Methods

The system design involves two processes: (i) the design of a robust system to produce the identifiers to prevent faults on the decodification of the package sent, and, (ii) the design of a hardware able to identify tags used from anywhere to recover the package.

A. Identification protocol

The redesign of the protocol identification during the multiplexing process prevents confusing bits of identification with a data packet. This error was found by comparing the

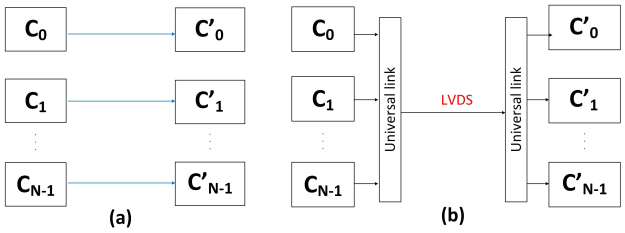

Fig. 1. Universal link for N different communications channels $(c_0, c_1, \cdots, c_{N-1})$ using LVDS. (a) N communication systems working individually. (b) Same system using an universal link. Each device is independent of the network configurations.

Fig. 2. Timing diagram of the universal link for inputs i from 0 to N. We show the new communication protocol including the initial '11' identifier.

displacement and the plot in different positions to which could represent an ambiguity with the original frame, and thus read a bad package during multiplexing [7]. To make the system robust, we add a general identifier composed of bits '11', ID (see Fig. 2). Which is added at the beginning of cycle-frame data at the serializer block. This new ID is imperative to obtain the correct frame.

B. Sampling frequency

The system arrangement of identification adds two bits at the top of each transmitted cycle. To compute the sampling frequency, we use

$$f_s \geq (N(\lceil \log_2 N \rceil + M) + 2) lcm\{f_i\}_{i=0}^{N-1}, \qquad (1)$$

where N is the number of inputs, M is the length of the bits in each package, f_i is the working frequency of each input signal, and lcm is the least common multiple of working frequencies of each input.

To avoid a fault during the demultiplexing process, the amount of bits by data package have to be limited up to a bit less than the longest identifier used in chain. Then, the maximum length of data package is calculated as:

$$M_{max} = \lceil \log_2 N \rceil + 1. \qquad (2)$$

C. Demultiplexer design

A general block diagram of the Universal Link transmitter and receiver hardware is showed in Fig. 3. The hardware dedicated to retrieving data consists of three blocks (see Fig. 3). The first one has the function to identify the beginning of each cycle with a constant '11' followed by the identifier of zeros. This tag (identifier) is compared with a main register that stores the data received from the multiplexer. Once this potential first identifier is found, let's call it 'beta' for now, the process continues to the comparison block, analyzing the following identifiers according to a predetermined logical order. Due to the dynamic nature of the compared register, it should not always wait until the new IDs are registered. After a failed comparison, raw data are stored in the registers and it should be compared while entering new data. Once the identifiers are found, and therefore the data sent, the demultiplexer is enabled and the data packages are recovered from the correct frame of information. Finally, each package of data is serialized.

These processes involve a delay to the output of the data. However, since the delay is constant in terms of number of clock periods, this does not affect the final transmission. The testing and implementation are conducted to determine the tolerance parameters for application and technical requirements for implementation. The delay is based on the number of signals N, the number of bits in each package M, and the specifications of the FPGA used.

D. Implementation of the full system

During the implementation of an arrangement for the operation, both blocks (see Fig. 3) were implemented in the same Atlys board [8]: the multiplexer (see [7]) and the demultiplexer. For a first test, they were internally connected between a serial signal from the multiplexer to the demultiplexer input. Thus, N inputs are defined from the board switches and output data can be viewed on the board LEDs. On this situation the real time constraints were not considered, but, although it was a single board implementation, delays processes took place due to the latency of each block and signal propagation.

For the final system, two subsystems are implemented in different Atlys boards, one for the multiplexer and another one for the demultiplexer. The signals to be tested are sent from one board (multiplexer) to the other board (demultiplexer) using a VHDCI Male-to-Male cable.

III. Results and Discussion

The initial implementation of the system was performed using an Atlys Spartan 6 board [8] for both the multiplexer and demultiplexer for the purpose of performance testing and decoding the multiplexed packets. After testing the performance for different input configurations and data sent, the system is implemented in different cards.

A. Multiplexer and demultiplexer simulation

The communication protocol based on identifiers was tested using the ISIM simulator[1] for a number of inputs from $N = 3$ to 8 signals. The working frequencies were 3 and 5 MHz for the input signal and according to (1), the sampling frequency f_s was set to 270 MHz. The demultiplexer for decoding showed a maximum of 200ns to find and extract data identifiers sent. In Fig. 4, the simulation shows that there is a constant delay in the transmission of 1360ns.

[1] Available online: http://www.xilinx.com/tools/isim.htm

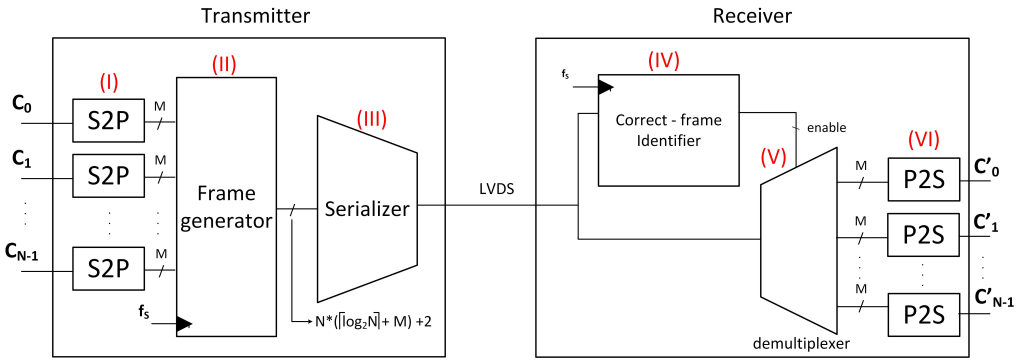

Fig. 3. Universal link general block diagram design: transmitter and receiver. Transmission process: (I) channel c_0 input signal is converted to M-bit parallel data. (II) The data package is arranged as a tag-based frame. Then is serialized (III) and sent. Demultiplexer has design three basic blocks: (IV) identifier, which analyzes and seeks the identification pattern, (V) demultiplexer, which breaks down the package so the data is sent to the output i depending on the label accompanying the data, and, (VI) the serial to parallel converter, which decomposes small vectors and serialized.

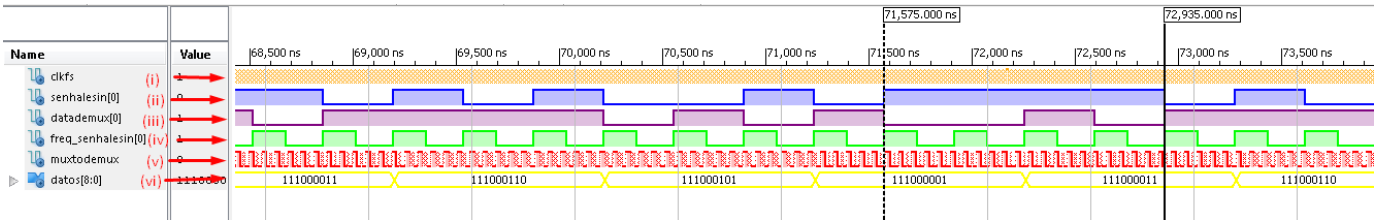

Fig. 4. Simulation of the signals of both communication blocks on the same board. (i) Clock of sampling frequency working at 270 MHz. (ii) Input signal connected to multiplexer. (iii) Recovered data from demultiplexing process. (iv) Working frequency of input signal connected to transmitter (5 MHz). (v) Bits sent through the LVDS connection. (vi) Data packages to be sent.

B. Resources and maximum frequencies: Multiplexer

Two approaches were used as guidelines for implementation: (i) time optimization and (ii) area optimization, focused in reducing the delay in processing the signal or the resources required in hardware, respectively. Table I shows the number of slice LUTs used to implement the multiplexer in the Xilinx Spartan 6 [9]. The tests were conducted for up to $N = 8$ inputs, each configuration operating with all possible quantities of bits per data packet based on (2).

Corresponding to the maximum frequency of the device according to that configuration, Fig. 5 shows the results based on the place and route phase step design implemented using the ISE design suite 14.7. We can clearly see that the maximum frequency does not has high variations depending on the number of input signals, as there is no difference in the number of data bits per packet increases. Even at the highest point of the graph, this frequency is maintained below the maximum permitted in the hardware used for implementation.

C. Resources and maximum frequencies: Demultiplexer

In the case of the demultiplexer block, the amount of resources used increased due to the Finder/Identifier block (see Fig. 3) as it is show in Table II compared to Table I as it can be seen in Fig. 6 . Also the maximum working frequency of the device does not change among several numbers of input signals or bits in the data package. Even at the highest working

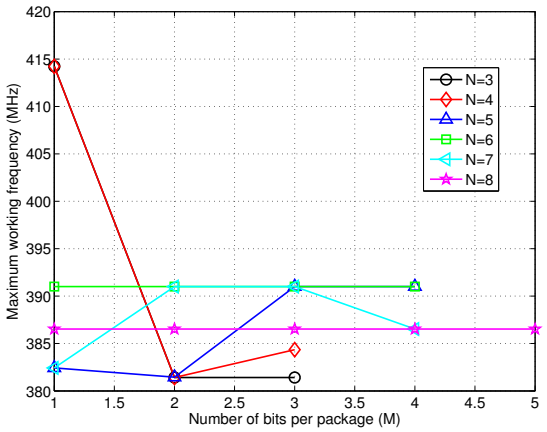

Fig. 5. Multiplexer: results of the maximum working frequencies at the Spartan 6 board Atlys optimized for minimum resources consumption with $N = 3 - 8$ and $M = 1 - 5$ for each case respectively.

TABLE I
NUMBER OF LUTS USED IN THE MULTIPLEXER. IMPLEMENTATIONS: (I) TIME OPTIMIZATION AND (II) AREA OPTIMIZATION.

	Time optimization				Area optimization			
N \M	1	2	3	4	1	2	3	4
3	20	21	22	-	18	21	22	-
5	21	24	27	31	23	24	25	27
7	25	26	33	37	24	25	26	31
8	19	25	34	38	25	28	29	39

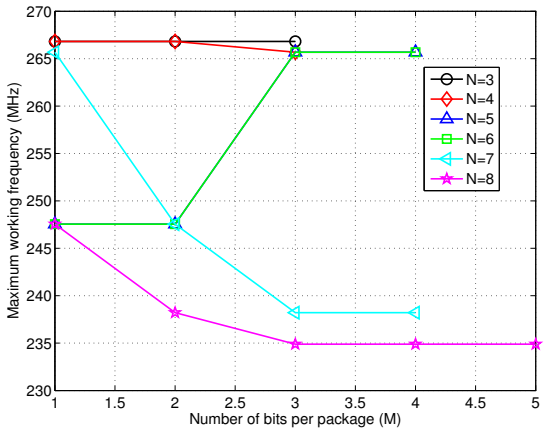

Fig. 6. Demultiplexer: results of the maximum working frequencies at the Spartan 6 board Atlys optimized for minimum resources consumption with $N = 3 - 8$ and $M = 1 - 5$ for each case respectively.

TABLE II
NUMBER OF LUTS USED IN THE DEMULTIPLEXER. IMPLEMENTATIONS: (I) TIME OPTIMIZATION AND (II) AREA OPTIMIZATION.

	Time optimization				Area optimization			
N \M	1	2	3	4	1	2	3	4
3	69	78	85	-	64	70	74	-
5	169	184	180	179	145	155	150	157
7	196	199	194	203	166	168	175	182
8	186	189	195	201	177	179	184	193

frequency, the device can still be successfully implemented in the Atlys Spartan 6 board.

D. First tests on a hardware implementation

We use a Universal Asynchronous Receiver/Transmitter (UART) component [10] as a first approach for testing the system design. Systematic tests were performed between the FPGA and the PC. The UART protocol provides 9 bits per package of sent data including 1 start bit and a byte of information. Those 9 bits were connected to the multiplexer input and then recovered at the demultiplexer to be sent back to the PC. For the test, the bit rate for the UART protocol was set to 9600bps. It is important to mention that this is not the final set of tests for the hardware implementation. Tests for higher speed communications rates and other testing methods for particular aspects in the design must be done for fully test of the system design.

The goal of working with the UART protocol set to 9600bps was to make a first hardware implementation. This time the tests were oriented to recover the information sent by the computer and then showed it back. Through extensive tests consisting of sending large strings of data, the device behaved as it was expected.

In the case of operating in two different Atlys boards connected by the LVDS VHDCI cable, a new state at the UART controller component had been created. It allows the transmitter to wait until the receiver had received, processed and sent the entire byte of information as the UART protocol required feedback between the transmitter and receiver components. After different tests, the time for processing the information would depend on the configuration and characteristics of the network devices; $20us$ were the time chosen to have enough time for the processing steps and continue the normal traffic of information. These tests using a UART interface does not allow us to test the system at full speed. However, it has been used as a proof of concept for this first hardware implementation.

IV. CONCLUSIONS AND FUTURE WORK

The new tag-based protocol for encoding the data and the identifiers was successfully tested for up to eight input signals at the multiplexer device. For the effective reconstruction of the serialized bits sent over the LVDS connection, a new relationship between the number of bits in each package and the number of input signals at the multiplexer was developed.

The multiplexer device and the demultiplexer device were implemented in two different Atlys boards. The input data at the multiplexer was successfully recovered at the receiver device and visualized at the output ports of the board.

Future work includes a testing of the LVDS tag protocol using the I2C protocol for a full duplex communications between two Atlys Spartan 6 boards.

REFERENCES

[1] S. Parkes, "High-speed, low-power, excellent emc: Lvds for on-board data handling," in *Proceedings of the 6th International Workshop on Digital Signal Processing Techniques for Space Applications, ESTEC, Sept.* Citeseer, 1998.

[2] S. Parkes and P. Armbruster, "Spacewire: a spacecraft onboard network for real-time communications," in *Real Time Conference, 2005. 14th IEEE-NPSS*, June 2005, pp. 6–10.

[3] S. Saponara, L. Fanucci, M. Tonarelli, and E. Petri, "Radiation tolerant spacewire router for satellite on-board networking," *Aerospace and Electronic Systems Magazine, IEEE*, vol. 22, no. 5, pp. 3–12, May 2007.

[4] D. Llamocca, V. Murray, Y. Jiang, M. Pattichis, J. Lyke, and K. Avery, "Scalable open-source architecture for real-time monitoring of adaptive wiring panels," *Journal of Aerospace Information Systems*, vol. 11, no. 6, pp. 344–358, 2014.

[5] V. Murray, D. Llamocca, J. Lyke, K. Avery, Y. Jiang, and M. Pattichis, "Cell-based architecture for adaptive wiring panels: A first prototype," *Journal of Aerospace Information Systems*, vol. 10, no. 4, pp. 187–208, Nov. 2013.

[6] S. Menon, Y. Bobra, A. Ghia, and A. Zaliznyak, "Programmable input/output circuit for fpga for use in ttl, gtl, gtlp, lvpecl and lvds circuits," Apr. 17 2001, uS Patent 6,218,858. [Online]. Available: http://www.google.com/patents/US6218858

[7] G. Patino, L. Sanchez, J. Lyke, and V. Murray, "Towards a fpga-based universal link for lvds communications: A first approach," in *XX International IBERCHIP Workshop*, Chile, Feb. 2014.

[8] Digilent. (2013, August) Atlys board reference manual. Digilent. [Online]. Available: http://www.digilentinc.com/data/products/atlys/atlys_rm_v2.pdf

[9] Xilinx. (2011, October) Spartan-6 family overview. Xilinx. [Online]. Available: http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf

[10] E. Bainville. (2013, April) Fpga simple uart. Bealto. [Online]. Available: http://www.bealto.com/fpga-uart.html