An FPGA-embedded oscilloscope based on the IEEE1451.0 Std.

Ricardo J. Costa[1], Diogo Eloi Pinho[1], Gustavo R. Alves[1] and Mário Zenha-Rela[2]
ISEP/CIETI/LABORIS[1] FCTUC/CISUC[2]
rjc@isep.ipp.pt, diogoeloi@hotmail.com, gca@isep.ipp.pt, mzrela@dei.uc.pt

Abstract

Digital oscilloscopes are adopted in several areas of knowledge, in particular in electrical engineering, since they are fundamental for measuring and classifying electrical signals. Thanks to the proliferation of Field Programmable Gate Arrays (FPGAs), embedded instruments are currently an alternative solution to stand-alone and modular instruments, traditionally available in the laboratories. High performance, low cost and the huge flexibility to change functional characteristics, make embedded instruments an emerging solution for conducting electrical experiments. This paper describes the project and the implementation of a digital oscilloscope embedded in a FPGA. In order to facilitate their control, an innovative architecture is defined according to the IEEE1451.0 Std., which is typically used to develop the denominated smart transducers.

Keywords: Embedded instrumentation, FPGA, IEEE1451.0 Std., Oscilloscope.

1. Introduction

In the last decades technology evolution has been changing the adopted instrumentation in electrical engineering laboratories. At the beginning analogue instruments were the only available solutions to measure or generate electrical signals. These instruments were typically large and heavy, providing a reduced number of features, which difficult a more detailed analysis of the electrical signals. It was difficult to observe particular phenomena and to get specific signals' characteristics, e.g. overshoots, average or rout mean square values, etc. Visual observations supported, when possible, with pen-and paper calculations were required.

Since the appearance of digital processors, new digital processing techniques emerged. The old analogue instruments are being replaced by digital ones, with all the signals defined in the digital domain. Nowadays, digital instrumentation allows gathering measurements in digital formats, facilitating, this way, the use of personal computers to characterize the measured signals.

Instrumentation is a reality in every laboratory. Besides a division in analogue or digital instruments, these last can be divided in the three main groups represented in figure 1, namely: i) stand-alone; ii) modular and iii) embedded.

Stand-alone instruments can accommodate both digital and analogue instruments, and have the entire architecture in a unique chassis that provides buttons and interfaces to control or observe the signals. Modular instruments are digital and they are mainly supported by a computer that controls dedicated hardware. The well none PXI system[1] can be integrated in this group, since the instruments (also named as virtual instruments) are provided by slot cards controlled by a computer, sometimes available in the same chassis. Embedded instruments also accommodate digital instruments and they can be seen as an alternative solution to stand-alone or modular. They comprise circuits implemented within chips for performing specific validation, test and debug functions of other electronic circuits in the same chip or circuit boards [1]. They are classified as a most cost-effective and flexible solution, since they are essentially supported by small devices such as µC, µP, FPAAs or FPGAs[2], this last well implemented in the market (Xilinx, Altera, etc.) and able to be easily reconfigured according to the measurement requirements of a particular circuit.

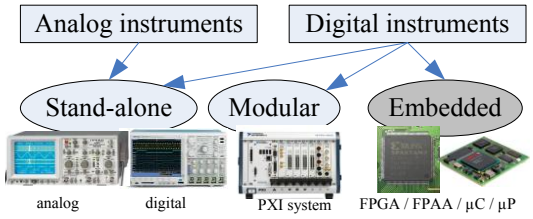

Figure 1: A possible classification of instruments.

It is precisely the enumerated advantages of embedded instrumentation with the flexibility provided by FPGAs, proved by the many projects

[1] PCI eXtensions for Instrumentation (PXI) is a modular instrumentation platform supported by the PXI Systems Alliance (http://www.pxisa.org/)

[2] µC (µController); µP (µProcessor); FPAA (Field Programmable Analog Array); FPGA (Field Programmable Gate Arrays).

using this type of devices (e.g. [2][3][4]), that incentivized the development of a digital oscilloscope embedded in a FPGA [5]. To complement some of the current research that is focusing on using the JTAG[3] interface, recently improved with the Internal JTAG interface (IJTAG)[4], to enable non-intrusive access and control of the embedded instruments, the described solution suggests the use of the IEEE1451.0 Std. A basic oscilloscope was therefore implemented in the core of an FPGA, following an architecture based on the IEEE1451.0 Std. The architecture uses a data structure named Transducer Electronic Data Sheet (TEDS) for describing and controlling all the features of the oscilloscope, and provides a set of IEEE1451.0 commands to enable its standard access and control.

Besides this introductory section, this paper has 5 other sections. Section 2 provides a brief overview about digital oscilloscopes, presenting their most common architectures. Section 3 suggests the use of FPGAs and the adoption of the IEEE1451.0 Std. for designing and accessing embedded instruments. Section 4 presents an implementation of a digital oscilloscope using an FPGA, and an architecture defined according to some issues of the IEEE1451.0 Std. Section 5 presents the verifications made to the implemented oscilloscope. Section 6 finalizes the paper with some conclusions.

2. Digital oscilloscopes

Digital oscilloscopes can be divided in: i) Mixed Signal Oscilloscopes (MSOs), ii) digital sampling oscilloscopes; iii) Digital Storage Oscilloscopes (DSOs) or; iv) Digital Phosphorus Oscilloscopes (DPOs) [7][8]. Since the last two are the most common in the market, it is reasonable to take a particular attention to them. As illustrated in figure 2, they use very similar architectures, differing in the adoption of a digital phosphor block.

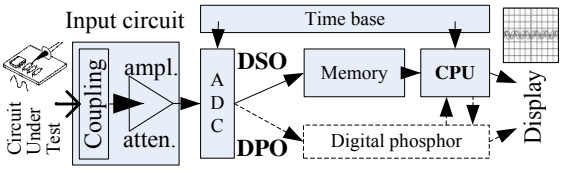

Figure 2: Block diagram of a digital oscilloscope according to the DSO and DPO architectures.

DSOs adopt serial architectures comprising an input circuit to acquire and handle analogue signals acquired from a Circuit Under Test (CUT). The input circuit is essentially implemented to remove (or not) the DC component, and to amplify/attenuate the amplitude of the measured signals according to the requirements defined by a particular user. The signals are then converted to the digital domain using an Analogue Digital Converter (ADC) using a sampling rate controlled by a time base block, which is defined by the users to adequately represent all signals in a display. Before that visual representation, the generated samples are processed using a Central Processing Unit (CPU) and gathered in a memory block, also controlled according to the defined time base. The CPU is the responsible to generate the samples of the measured signal to be visualized in the display.

DPOs follow the same architecture, but comprise an additional parallel block named digital phosphor. This block is essentially used to facilitate displaying the samples, freeing up the memory and the CPU for other computational tasks, namely for running specific digital calculations (digital filtering, interpolations, triggering, etc.). In comparison to the DSOs, this type of architecture brings additional advantages to the signal representation. It allows managing the intensity of all samples, providing more information about the signal, and provides higher bandwidth, since the sampling rate is faster, which facilitates the capture of sporadic and very fast events on a signal (e.g. signal glitches).

Based on the simpler DSO architecture, a digital oscilloscope described for an FPGA and defined according to some issues of the IEEE1451.0 Std., can be implemented as an embedded and "intelligent" oscilloscope.

3. Using FPGAs and the IEEE1451.0 for designing embedded instruments

The processing required to handle samples in digital instruments, in particular in an oscilloscope, can be implemented by several devices, such as µP, µC, FPAAs or FPGAs. Despite the last two allow the reconfiguration of hardware, FPAAs still have a limited number of analogue components in their core, not providing the same flexibility as FPGAs that comprise many digital blocks. Therefore, FPGAs are preferable for implementing part or the entire architecture of one or more instruments, since they are hardware reconfigurable using standard Hardware Description Languages (HDLs), and they can run several hardware blocks in parallel. This parallelism offered by FPGAs simplifies the implementation of specific digital algorithms required to handle the samples of a particular

[3] Joint Test Action Group (JTAG) is a common name for what later became the IEEE1149.1 Std. That stands for Standard Test Access Port and Boundary-Scan Architecture [6].

[4] Internal Joint Test Action Group (IJTAG) is an interface standard to instruments embedded in chips that defines a methodology for their access, automating their operations and analyzing their outputs. It is currently defined by the IEEEp1687 Std. (http://standards.ieee.org/develop/wg/IJTAG.html).

oscilloscope, and allows running more than one instrument in the same core.

Despite most of the processing is made in the digital domain (e.g. interpolation techniques, implementation of filters, etc.), providing flexibility in the design of a particular instrument, they must provide interfaces to the analogue domain. Therefore, analogue to digital conversions are required, as well as power drivers' interfaces to interconnect the CUT. For this purpose, the use of FPGA-based boards for accommodating the entire architecture of an oscilloscope, or any other instrument, is an interesting solution. The core of the instrument can be embedded into an FPGA, while other issues, such as memories, the amplification/attenuation of signals, among others, can be supported by the surrounding devices available on those boards, as represented in figure 3.

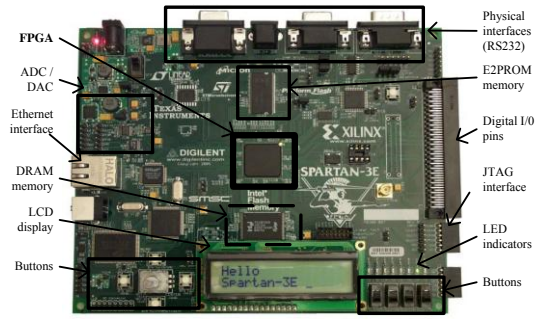

Figure 3: Example of a typical FPGA-based board.

Since the core of an FPGA is reconfigurable, it is easy to change the functionalities of a particular instrument by reconfiguring the hardware modules described in HDL. Unlike the use of µPs or µCs, whose programming depends on the assembly language provided by the manufacturer, FPGAs can use the same HDL code to reconfigure different instruments. Despite Verilog and VHDL are standard HDLs, they do not guarantee the standard access of a particular instrument embedded into an FPGA. For this purpose, using the IEEE1451.0 Std. is a solution, since it describes a particular architecture to manage the so-called smart transducers, whose architecture can be adopted to design an embedded instrument, and, in particular, an oscilloscope.

Defined in 2007, the IEEE1451.0 Std. [9][10] aims to network-interface transducers (sensors and actuators) and defines a set of operating modes, based on specifications provided by Transducer Electronic Data Sheets (TEDSs). This standard specifies operating modes controlled using commands that can be applied through a set of APIs. As represented in figure 4, it defines an architecture based on two modules that should be interconnected using an interface protocol: the Transducer Interface Module (TIM) and the Network Capable Application Processor (NCAP). This last provides remote access to the Transducer Channels (TCs) and to the TEDSs, both traditionally included in the TIM. A TIM may integrate several TCs (up to 65535), defined as the digital transducers. Each module is connected through an interface defined by another standard of the IEEE1451.x family, some already specified according to the IEEE1451.0 Std. (e.g. the IEEEp1451.6 Std. for the CANopen interface) and others intended to be modified in the future (e.g. IEEE1451.2 Std. which defines point-to-point interface).

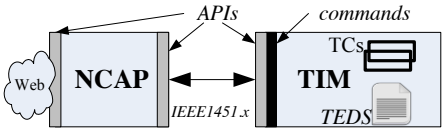

Figure 4: Overall architecture of the IEEE1451.0 Std.

Despite the relevance of the NCAP in the entire architecture, the TIM is the main module and the one that traditionally implements the hard-processing of any smart transducer. Therefore, after a careful analysis on the TIM's characteristics and on the requirements posed by a digital oscilloscope, it was decided to embed the oscilloscope within the FPGA as a TIM. The commands provided by the TIM allow the access and the control of the oscilloscope, whose definitions can be made through an internal TEDS constructed according to a particular structure. Using a simple computer interface can therefore enable the control of the oscilloscope and the representation of the measured signals.

4. The implemented oscilloscope

To prove the possibility of using the IEEE14151.0 Std. with FPGA devices to create an embedded instrument, a digital oscilloscope prototype was conceived and implemented according to a DSO architecture. Some specifications of the IEEE1451.0 Std. were adopted to define and control the entire behavior of the oscilloscope, namely the use of a TEDS and a set of commands. As illustrated in figure 5, the oscilloscope was implemented in an FPGA-based board from Xilinx (XC3S700AN starter kit) to acquire signals from a CUT (emulated by a traditional function generator), using an internal ADC. Through an RS-232 serial connection, the oscilloscope was attached to a computer running an interface developed in JAVA, represented in figure 6. This interface allows users to access and control all the functionalities of the oscilloscope.

The oscilloscope was implemented as the TC number 1 within a TIM running inside the FPGA. The TIM and the associated TC operate using modes defined in the IEEE1451.0 Std. The TIM can run in the *Initialization* or *active* modes, while the TC can run in the *Initialization*, *Inactive* or *Operational* modes. The transactions among the different operational modes are established through internal operations of the oscilloscope or through commands issued by the users.

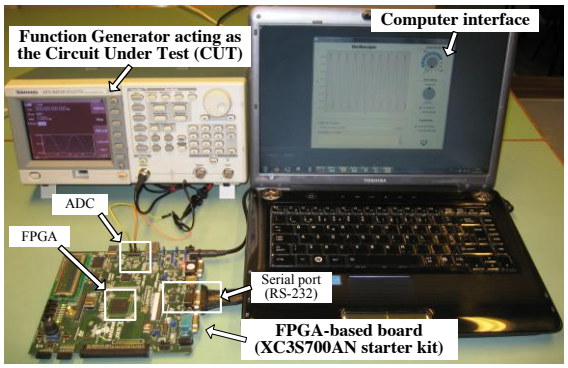

Figure 5: Picture of the implemented oscilloscope.

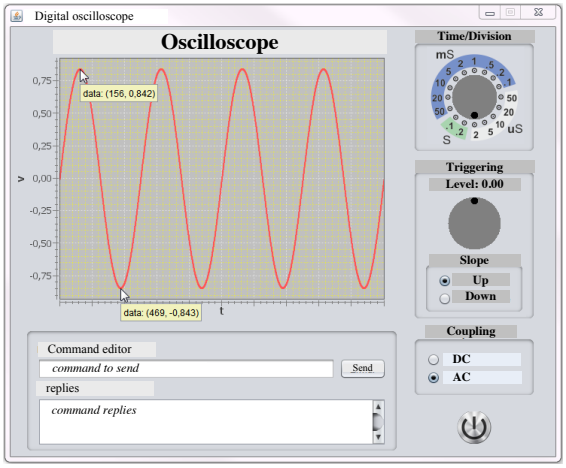

Figure 6: Computer interface used to control the digital oscilloscope.

The TIM is the main unit of the entire oscilloscope that was implemented in hardware blocks described in the Verilog HDL. As illustrated in the diagram of figure 7, it is divided three main modules: i) a Central Processing Unit (CPU); ii) a reformulated TC-TEDS and; iii) a Communication Module (CM).

The CPU is the unit that handles all samples used to display in the computer interface the measured signal. As represented in figure 8, after sampling the signal, the CPU defines the coupling mode (AC or DC) and implements interpolations and triggering methods to fill-in a data set with 2500 samples used to represent the signal. The number of samples was empirically selected, in order to get a good visual representation of the signal.

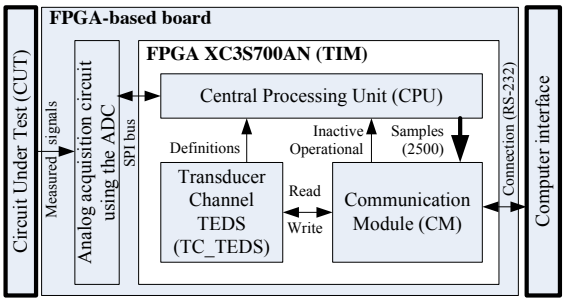

Figure 7: Architecture of the implemented oscilloscope.

The selected hardware to acquire the signals, namely the ADC and the FPGA maximum operation frequencies, limited the sampling rate up to approximately 1.5MHz[5], and to a bandwidth of 575 kHz calculated base on a step response of the oscilloscope, as it will be described in the next section 5 of this paper. Additionally, the amplitude of the measured signal is limited, going from 0.4 V up to 2.9 V due to hardware constrains posed by the FPGA-based board.

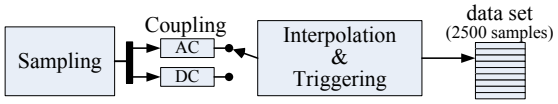

Figure 8: Picture of the implemented oscilloscope.

The coupling allows removing (or not) the continuous component of an analog signal by selecting the DC or AC modes. This functionality was implemented using a digital high pass band Finite Impulse Response (FIR) filter. This filter was projected using the Matlab software according to the Parks-McClellan algorithm[6] [11]. The filter was calculated with 318 coefficients with a normalized pass band frequency of 0.03; normalized cut-off frequency of 0; maximum deviation on the pass band of 0.001dB; and a minimum rejected band of -80dB. The normalized response of the implemented filter is represented in figure 9.

[5] The ADC available in the FPGA-based board is the LTC 1407A-1, which requires a minimum of 34 clock cycles to acquire a sample. Since it is controlled by the FPGA available in the board, that runs using a digital clock of 20ns, the maximum acquisition frequency (fa) is approximately 1.5 MHz [fa_{max}=1/(20ns x 34)=1.47MHz].

[6] http://www.mathworks.com/help/signal/ref/firpm.html; http://www.mathworks.com/help/signal/ref/firpmord.html

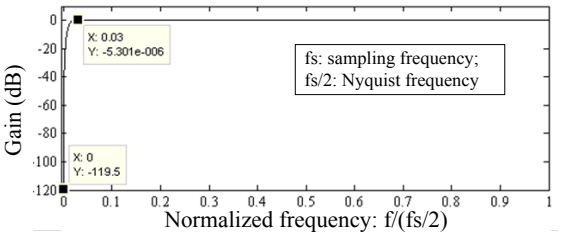

Figure 9: Response of the implemented filter to cut-off the direct current of the measured signal.

Independently of the selected coupling mode (AC or DC), the remaining circuit must generate 2500 samples to represent the measured signal in the computer interface. Those samples are generated by the CPU using an interpolation factor (L) and by the level and slope defined for triggering the signal. The interpolation factor is calculated according to several interrelated parameters defined in table 1, namely: i) the measured time (mt), which may be seen as a time window used to capture a signal defined by the user in the computer interface through the time/division knob button; ii) the number of acquired samples (as), internally determined by the CPU; iii) the selected sampling frequency of the ADC (fs), and; iv) the required frequency (fr) to get the indicated 2500 samples. For better understanding the entire generation process of the 2500 samples, figure 10 and equations 1, 2 and 3 represent these variables and their inter-reliance.

Table 1: Parameters that influence the process of filling-in the data set with 2500 samples.

Measure time (mt)	Required freq. (fr)	Samp. freq. (fs)	Samples acq. (as)	Interpol. factor (L)
20 µs	125 MHz	1.25 MHz	25	100
50 µs	50 MHz	1.25 MHz	62.5	40
100 µs	25 MHz	1.25 MHz	125	20
200 µs	12.5 MHz	1.25 MHz	250	10
500 µs	5 MHz	500 kHz	250	10
1 ms	2.5 MHz	250 kHz	250	10
2 ms	1.25 MHz	125 kHz	250	10
5 ms	500 kHz	50 kHz	250	10
10 ms	250 kHz	25 kHz	250	10
20 ms	125 kHz	12.5 kHz	250	10
50 ms	50 kHz	5 kHz	250	10
100 ms	25 kHz	2.5 kHz	250	10
200 ms	12.5 kHz	1.25 kHz	250	10
500 ms	5 kHz	500 Hz	250	10
1 s	2.5 kHz	250 Hz	250	10
2 s	1.25 kHz	125 Hz	250	10

The interpolation was simulated in Matlab and implemented using Cascaded Integrator Comb (CIC) interpolators balanced with FIR filters [12][13]. The result was two interpolation factors (L_{FIR} and L_{CIC}) whose product results in the referred L factor previously indicated to calculate the required frequency (fr) to generate the 2500 samples. The selected triggering is used to define the time when the first sample is acquired in the time window defined by the mt parameter.

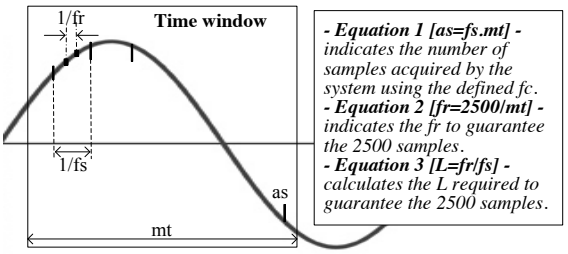

Figure 10: Example of sampling and interpolating a signal and associated equations.

Besides all these digital processing methods that may be encountered in any digital oscilloscope, the innovation of the implemented solution focuses on the use of the IEEE1451.0 Std. Therefore, to enable users to get real time information about the oscilloscope, and to control its functionalities, a TEDS was defined and implemented within a memory block of the FPGA. This TEDS was designed based on the structure of a TC TEDS defined by the IEEE1451.0 Std. It is divided in different blocks and fields of eight bits (octets), organized according to a Type-Length-Value (TLV) structure, as represented in figure 11. The TEDS's structure comprises three distinct blocks: i) length (2 octets), indicating the number of octets available in the remaining blocks; ii) data block, providing the main information about the oscilloscope in 12 TLV structures with a total of 107 octets and; iii) a checksum field used to verify the data integrity.

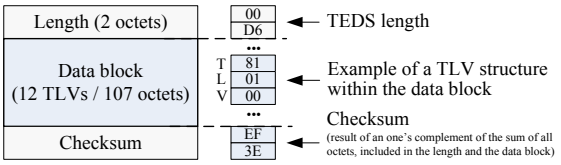

Figure 11: Implemented TEDS structure.

Each TLV indicates different issues, such as: i) defines the oscilloscope as a sensor; ii) the units retrieved from the oscilloscope (volts); iii) the minimum and maximum amplitude of a signal able to measure (0.4 V to 2.9 V); iv) measurement resolution (0.15 mV defined according to the ADC); v) the number of samples available in the data set (2500) and their representation format (float - 4 octets); vi) trigger level and slope; vii) the way samples are transmitted to the computer interface, which was defined in an *immediate mode*, indicating that the samples in the data set are immediately transmitted to the computer when it is full and; viii) scale definitions for representing the samples in the

computer interface (the size of the provided display window).

Using the information provided within the TEDS data block, the CPU controls all the oscilloscope operations. When in the *active mode* it reads the TEDS's fields and, based on its values, controls all the operations of the oscilloscope to fill-in the data set with the 2500 samples. When it is filled-in, all samples are sent to the computer interface to represent the measured signal, according to a specific field defined in the TEDS that indicates the adoption of an immediate transmission of all data to the computer.

Therefore, the control and characteristics of the oscilloscope are defined within the TEDS's fields and managed by the computer interface through some IEEE1451.0 commands. These commands, and associated replies, are defined according to the IEEE1451.0 standard message structures divided in the octets illustrated in figure 12.

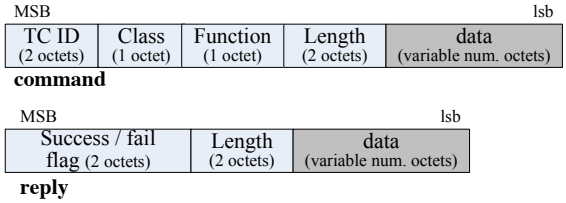

Figure 12: Message structures used for commands and associated replies.

As discriminated in table 2, four commands were implemented: i) *TCoperate*; ii) *TCidle*; iii) *readTEDS* and; iv) *writeTEDS*. All these commands follow the same structure defined in the standard. The first two are adopted to place the oscilloscope in the *operation* or *idle* modes, which means turning it on or off. The *read/write TEDS* commands are adopted to control the oscilloscope. They are sent through the computer interface when a user changes a specific button or when they manually define a particular command using the command editor. Every issued command to the TIM generates a particular reply described in table 3, indicating a successful or a faulty operation. The replies generated follow the same structure of the IEEE1451.0 Std., but most of them add particular codes´ responses, enumerated in table 4, for better management of the oscilloscope using the computer interface. Internally, it is the CM that decodes commands received from the computer interface and creates and sends the associated replies. This way, it is possible to control the oscilloscope in a standardized way (using IEEE1451.0-based commands) and to provide information about it.

Table 2: Implemented commands' data structures.

command	TC	class	func.	length	data	
					TEDS	data
TCoperate	01	04	01	00.00	-	
TCidle	01	04	02	00.00	-	
ReadTEDS	01	01	02	00.05	03	p
WriteTEDS	01	01	03	Δ	03	p+d

Δ-variable length; p- offset position (octet location to read/write from/to the TEDS); d-data (data to write in the TEDS).

All the functionalities of the oscilloscope and some specific aspects were verified using a particular scenario that includes monitoring the messages and associated replies between the computer interface and the TIM, and measuring some representative signals, namely a step signal and some sinusoidal waves with different frequencies.

Table 3: Data structures of the commands' replies and the data transmitted to the computer interface.

Response to	Succe. (01) Fail (00)	Length (2 octets)	Data response
Error message	00	00.01	C[00-06]
WriteTEDS; TCoperate/idle	01	00.01	C07
ReadTEDS	01	Δ	C08 + Δd
Data sent to the computer interface	01	27.10 (hex)	10000*

Δ-variable length of the retrieved TEDS (depends on the op defined in table 2); Δd-data retrieved from the TEDS, Cx- indicates reply codes indicated in table 4; *in float format (4 octets per each sample, totalizing 10000/4=2500 samples).

Table 4: Codes' responses to issued commands.

Lost message header	C00
Invalid TC	C01
Unknown class/function ID	C02
Lost command	C03
Unknown TEDS ID	C04
Invalid TEDS position	C05
Checksum error	C06
Successful write operation	C07
Read response	C08

(note: the values in the table are in the hexadecimal format)

5. Verification of the oscilloscope

To verify the correct operation of the implemented oscilloscope, a scenario was defined according to the schematic of figure 13. It uses the implemented oscilloscope (FPGA-based board connected to a computer), and two stand-alone instruments, namely a function generator[7] and a digital oscilloscope[8]. The function generator was adopted to emulate a common CUT, generating different signals able to be measured by the implemented oscilloscope. The stand-alone oscilloscope was adopted for comparison purposes, i.e. to verify if the implemented oscilloscope behaves in a similar way as the commercial one.

[7] AFG 3021B from Tektronix.
[8] TDS 3114b from Tektronix

The correct operation and some characteristics of the implemented oscilloscope was verified in different stages, namely: i) by monitoring commands exchanged between the FPGA-based board and the computer interface using a serial monitor application installed in the computer and; ii) by measuring different signals generated by the stand-alone function generator.

Figure 14 exemplifies the observation made to the command messages and associated replies when a specific button is changed or a particular command is edited in the computer interface (in the particular case the *WriteTEDS* command).

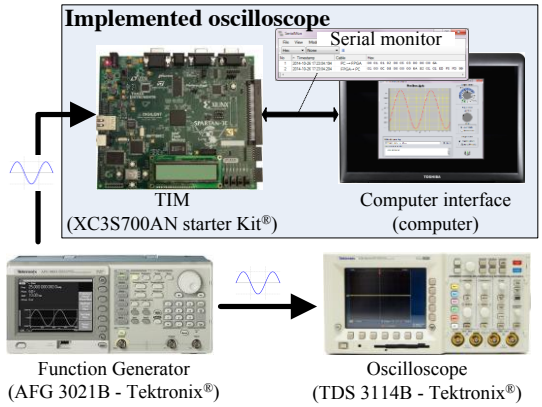

Figure 13: Adopted scenario to verify the oscilloscope operation.

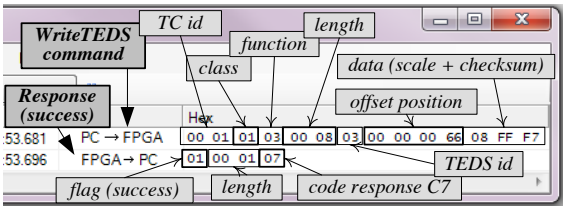

Figure 14: Example of a WriteTEDS command and associated reply monitored by the serial monitor.

To get particular characteristics of the oscilloscope, in particular its bandwidth, a step signal was applied and analyzed by the oscilloscope, as represented in figure 15. The step was defined with 1V of amplitude (from 1V to 2V) and a rise time of 1.8ns. A scale of 2µs that is associated to the scale of 20 µs in table 1[9], was selected in the control interface, which means a sampling frequency of 125MHz (fr). The time between two consecutive samples is therefore 8ns, which means a rising time (tr) of 0.608µs that defines a bandwidth of 575kHz, calculated according to the indicated equation retrieved from [14].

[9] Each time/division in the computer interface is associated to the scales indicated in table 1 attenuated by a factor of 10.

While the step was essentially adopted to understand the dynamic response of the oscilloscope, sinusoidal signals were also applied to both oscilloscopes for comparison purposes. It was selected three sinusoidal signals with different amplitudes, frequencies and DC components, as detailed in table 5. The range values of the selected signals were defined according to the characteristics of the implemented oscilloscope, namely the bandwidth and the amplitudes it can measure.

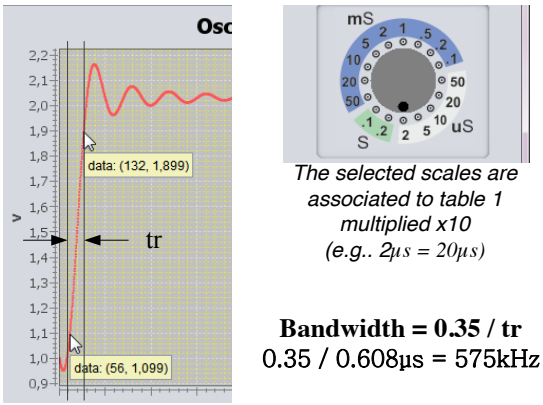

Figure 15: Step response of the oscilloscope.

Table 5: Sinusoidal signals applied to the oscilloscopes.

frequency	amplitude	DC component
500kHz	2V	1.65V
100kHz	1V	1V
1Hz	2.4V	1.65V

A detailed comparison of all signals in both oscilloscopes' displays was made, leading to conclude the correct operation of the implemented oscilloscope, since all signals were represented similarly in both displays, as exemplified in figure 16 for the signal with 100kHz.

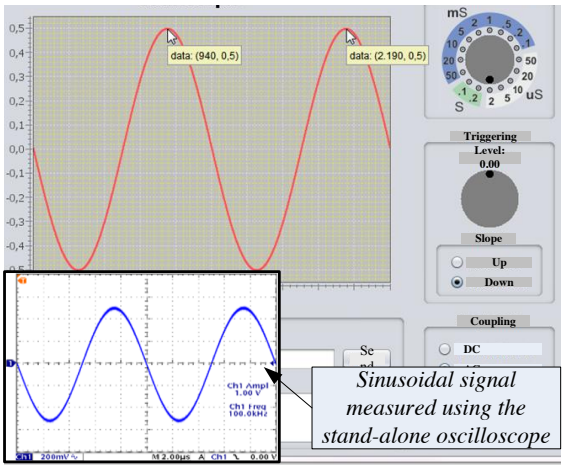

Figure 16: Example of a measurement of the sinusoidal signal with 100kHz with the implemented and the stand-alone oscilloscopes.

6. Conclusions

The technology evolution of the last two decades and the emergence of new and sustainable digital processing techniques, incentivized a shift on the type of adopted instruments in any laboratory. The old analogue instruments are being replaced by digital ones, since they provide more reliable results and offer the possibility of handling data using personal computers. The traditional and modular instrumentation are very common today, providing different types of computer interfaces. However, their adoption to measure/generate particular signals from/to a CUT can be prohibitive, due to development costs. The evolution of small devices, namely μP, μC, FPAAs and in particular FPGAs, are incentivizing the appearance of the so-called embedded instrumentation. Despite these instruments are essentially adopted to test internal circuits within the same board or chip, they can also be seen as traditional instruments able to control using a personal computer. The advantage is the possibility of easily reconfiguring their functionalities without changing the entire platform. This way, the proliferation of FPGAs in the market, and the capability of reconfiguring their internal hardware blocks with modules developed using standard HDLs, such as Verilog or VHDL, make this type of devices the most appropriated for developing embedded instrumentation.

The project described in this paper proposed the use of a FPGA to embed a digital oscilloscope described in Verilog and externally controlled through a personal computer. Additionally, since there is a lack of a standard for controlling and accessing a similar solution, it was suggested the use of some issues of the IEEE1451.0 Std., namely a TEDS and a set of commands. Although the implemented oscilloscope is a prototype, which would require some improvements concerning its robust operation, the developments described and the obtained results, led to conclude the real advantage of using the IEEE1451.0 Std. and FPGA technology for developing embedded instruments able to be externally controlled as traditional stand-alone or modular instruments. The use of standard commands to control their operation and to get real-time information from their associated TEDS, was seen as an interesting and promising solution to take into consideration for the development of other instruments embedded in FPGAs. The design of embedded instruments based on reconfigurable technology, such as FPGAs, and supported on the IEEE1451.0 Std., may contribute for the proliferation of those instruments to run the traditional experiments in laboratories, replacing the traditional stand-alone and modular instrumentation.

Note: This paper resumes a MSc. thesis submitted by the second author to the Polytechnic Institute of Porto – School of Engineering (IPP/ISEP) - Portugal.

References

[1] Frost & Sullivan - White Paper, "Embedded Instrumentation Its Importance and Adoption in the Test & Measurement Marketplace." http://www.frost.com, 12-May-2012.

[2] Fernando Pereira, Luís Gomes and Luís Redondo, "FPGA Controller for Power Converters with integrated Oscilloscope and Graphical User Interface," in *Proceedings of the 2011 International Conference on Power Engineering, Energy and Electrical Drives*, 2014, p. 6.

[3] B.Hemalatha, V.R.Ravi, S.Divya and M.Uma, "Embedded FPGA Controller for Robot Arm in Material Handling Using Reconfigurable NICompact RIO," in *2 nd International Conference on Current Trends in Engin eering and Technology, ICCTET'14*, 2014, p. 7.

[4] Fernando Machado et al., "FPGA-Based Instrument for Satellite Beacon Monitoring on Propagation Experiments," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 12, p. 10, Dec. 2014.

[5] Diogo Pinho, "MSc. Thesis - Projeto e implementação de osciloscópio digital baseado na norma IEEE1451.0," *Inst. Super. Eng. Porto ISEP*, p. 152, 2014.

[6] IEEE1149.1-2013™, "IEEE Standard for Test Access Port and Boundary-Scan Architecture," *Inst. Electr. Electron. Eng. Inc*, p. 442, May 2013.

[7] A. Campilho, *Instrumentação electrónica: métodos e técnicas de medição, 1ª ed.* FEUP edições, 2000.

[8] Tektronix, *XYZs of Oscilloscopes*. available in: http://info.tek.com/rs/tektronix/images/03W_8605_6.pdf, 2011.

[9] IEEE Std. 1451.0™, "IEEE Standard for a Smart Transducer Interface for Sensors and Actuators - Common Functions, Communication Protocols, and Transducer Electronic Data Sheet (TEDS) Formats," *Inst. Electr. Electron. Eng. Inc*, p. 335, Sep. 2007.

[10] Eugene Y. Song and Kang Lee, "Understanding IEEE 1451-Networked smart transducer interface standard What is a smart transducer," *IEEE Instrum. Meas. Mag.*, pp. 11–17, Apr. 2008.

[11] J. H. McClellan and T. W. Parks, "A personal history of the Parks-McClellan algorithm," *IEEE Signal Process. Mag.*, vol. 22, no. 2, pp. 82–86, Mar. 2005.

[12] L. Milic, *Multirate Filtering for Digital Signal Processing: MATLAB Applications*. IGI Global, 2009.

[13] E. Hogenauer, "An economical class of digital filters for decimation and interpolation," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 29, no. 2, pp. 155–162, Apr. 1981.

[14] Tektronix, "Technical brief: Understanding Oscilloscope Bandwidth, Rise Time and Signal Fidelity." 2008.