SoC-FPGA systems for the acquisition and processing of electroencephalographic signals

Matías Javier Oliva, Pablo Andrés García, Enrique Mario Spinelli, Alejandro Luis Veiga
Biomedical Industrial and Scientific Instrumentation group (GIBIC), Electronic Control and Signal Processing Investigation Institute (LEICI), Engineering Faculty, La Plata National University (UNLP), Buenos Aires, Argentina

Article Info

Article history:

Received Aug 11, 2021
Revised Sep 29, 2021
Accepted Oct 19, 2021

Keywords:

Biopotentials
Brain-computer interfaces
Digital systems design
SoC-FPGA systems
Steady-state evoked potentials

ABSTRACT

Real-time acquisition and processing of electroencephalographic signals have promising applications in the implementation of brain-computer interfaces. These devices allow the user to control a device without performing motor actions, and are usually made up of a biopotential acquisition stage and a personal computer (PC). This structure is very flexible and appropriate for research, but for final users it is necessary to migrate to an embedded system, eliminating the PC from the scheme. The strict real-time processing requirements of such systems justify the choice of a system on a chip field-programmable gate arrays (SoC-FPGA) for its implementation. This article proposes a platform for the acquisition and processing of electroencephalographic signals using this type of device, which combines the parallelism and speed capabilities of an FPGA with the simplicity of a general-purpose processor on a single chip. In this scheme, the FPGA is in charge of the real-time operation, acquiring and processing the signals, while the processor solves the high-level tasks, with the interconnection between processing elements solved by buses integrated into the chip. The proposed scheme was used to implement a brain-computer interface based on steady-state visual evoked potentials, which was used to command a speller. The first tests of the system show that a selection time of 5 seconds per command can be achieved. The time delay between the user's selection and the system response has been estimated at 343 µs.

Corresponding Author:

Matías Javier Oliva
Biomedical Industrial and Scientific Instrumentation group (GIBIC)
Electronic Control and Signal Processing Investigation Institute (LEICI)
Engineering Faculty, La Plata National University (UNLP)
Calle 1 y 47, La Plata, Buenos Aires, Argentina
Email: matias.oliva@ing.unlp.edu.ar

1. INTRODUCTION

System on a chip field-programmable gate arrays (SoC-FPGA) are devices that combine in a single chip a field programmable gate array (FPGA) and a general-purpose processor. The high parallelism capacity of FPGAs has placed them as a fast and economical option, halfway between application-specific integrated circuit (ASIC) devices and general-purpose processors [1], outperforming even digital signal processing (DSP) devices when real-time requirements become more demanding [2]. On the other hand, having a general-purpose processor allows designers to implement a simple and intuitive user interface, alongside with any other high-level tools required, overcoming the limitations of FPGAs in this regard. By combining these two processing elements, SoC-FPGA devices have become a very good option for the development of systems where real-time processing is key.

The acquisition and real-time processing of biological signals, particularly electroencephalographic (EEG), has promising applications in the implementation of brain-computer interfaces (BCIs). These are devices that provide the user with a new channel of communication and control without performing motor actions [3], which can be very useful for people with reduced mobility and loss of speech to have the opportunity to command a speller [4], a wheelchair or a computer mouse, for example [5], [6].

In order to achieve this goal, BCIs acquire brain potentials from the user, commonly by EEG. The EEG record suffers disturbances, well documented in the bibliography, in the face of different stimuli, which can be external or internal to the user. By processing it, it is possible to detect in real time the disturbances related to the stimuli, and to use this information to control different devices. There are many documented brain potentials, so there is a variety of BCI systems, depending on the potential they acquire and process. Some of the most commonly used are: slow cortical potentials [7], motor imagery [8], event-related synchronization and desynchronization (ERS/ERD) [9] and visual evoked potentials. These last ones can be classified as steady-state (SSVEPs) and transient state (TVEPS) [10] potentials. Most of the current research on the subject makes use of a biopotential acquisition system and a personal computer (PC). This greatly limits the portability and flexibility of the set, so it is time to migrate to an embedded system, eliminating the PC from the scheme. The design of BCIs according to the different paradigms has substantial differences, but several aspects in common: all of them require real-time acquisition and processing of EEG signals, high computation speed, so that the user has fast feedback to its commands, and an intuitive and easy-to-use user interface. For these reasons the authors believe that SoC- FPGAs have a lot to contribute to the design of BCI systems.

Particularly, SSVEPs are periodic potentials that can be obtained by EEG recording in the occipital area. They appear when the user is presented with periodic visual stimuli above 6 Hz, and present the same periodicity as the stimulus. SSVEP based BCIs are among the ones that can achieve the highest information transfer rate (ITR) [4]. They also have the advantage of being simple to use devices, that require very little training, hence they have been widely studied. Examples of these implementations can be found at: [4], [5], [11]-[13]. These types of BCIs impose very demanding time constraints, such as the generation of several visual stimuli that have to be synchronized with the signal's acquisition. Its implementation with a microcontroller porting a real time operation system (RTOS), for example, can produce a dispersion on the latency, introducing phase noise. The FPGAs can address this issue, synchronizing both the acquisition and the stimuli with a single clock, resulting in very low and constant latencies.

The implementation of SSVEP-based BCI systems in FPGAs is a relatively new subject of study, but has given rise to publications such as [14], where a SSVEP-based BCI with phase encoding that can achieve a 20 bits/min ITR (5 to 8 seconds per selected command) is presented, or [15], which reports an ITR of 36 bits/min. Although these devices do not reach the ITRs achieved by the systems reported in [4] or [13], they do stand out for their portability and flexibility. None of these works uses an architecture like the one proposed, which combines the computational speed of an FPGA with a general-purpose processor for high-level tasks in a single chip.

This article discusses the use of SoC FPGA systems to acquire and process EEG signals. The advantages of using this type of device are evaluated, and it is discussed how the different processing elements can be used for each of the tasks involved. Afterwards it is shown the development of a SSVEP based BCI, with frequency encoding, developed entirely on a de10-nano board, based on a Cyclone V SoC-FPGA provided by Intel Altera, and its use to command a speller, well suited for the Spanish language. As a result, a device that stands out for being compact, flexible, portable, low-cost, easy to use and independent from a host PC for its operation, is obtained. Although the implemented system is based on SSVEPs, a large part of the tools developed can be used to implement different types of BCIs.

2. MATERIALS AND METHODS

2.1. System on chip

When developing high-quality real-time systems, developers must choose between two families of digital devices. The first option is a pure-software approach, where a microprocessor, used as the core, communicates with a DSP or other external devices, forming a multiple-chip system. The second option consists of a system on chip (SoC), in which all the necessary electronic circuits for a given system are integrated in a single circuit (IC). The result is a device that uses less energy, has better performance, requires less space, achieves a higher communication bandwidth between the different subsystems, and is more reliable than systems that involve multiple chips [1].

The combination of a microprocessor and an FPGA is the core of many embedded systems. As mentioned, there are plenty of reasons to integrate them, along with different peripherals that are of interest, in a single chip. In this way, it is possible to integrate the high-level functionalities of a processor with the real-time operation of an FPGA, ensuring an optimal interconnection between both processing elements.

In these heterogeneous systems, FPGAs are ideal for handling parallel operations of many data channels; and, because they implement computation directly in hardware, they provide a low and constant latency path for tasks such as custom triggering and high-speed closed-loop control. On the other hand, they improve the flexibility of embedded systems, making them easier to update than systems with fixed logic and allowing them to adapt to changing I/O requirements. In this scheme, the designers can solve the tasks that demand low latency using the FPGA, while the embedded processor takes care of the user interface and the rest of the tasks with lesser time constraints, possibly porting an operating system for it.

Currently there are two manufacturers that concentrate almost the entire offer of SoC-FPGA systems. These are Xilinx and Intel (formerly Altera). The de10-nano board, selected for this article, features a Cyclone V 5CSEBA6U23I7 FPGA-SoC, part of the low-range of Intel Altera devices. This chip combines the Cyclone V FPGA with an ARM-Cortex A9 processor, which carries an Angstrom distribution of a Linux operating system, alongside with interconnection buses between them. Additionally, the board provides a LTC 2308 chip: a 500 ksps, 12-Bit, successive approximation register (SAR) analog-to-digital converter (ADC). The 110 K logic elements of the Cyclone V 5CSEBA6U23I7 device are more than enough for the type of processing required for this application, as it is to be shown in section 3.1, and the ADC included in the board is well suited for the acquisition of the EEG signal, hence the selection of the de10nano board constitutes a cost-effective solution. In Intel Altera's documentation the microprocessor is often referred to as a hard processor system (HPS). This acronym will be used on this paper as well.

2.2. BCI implementation using SoC

In order to implement a BCI device it is necessary to obtain electroencephalographic signals from the user and process them in real time, in the shortest possible time. The processed information must be classified by some criteria and then used to control a device. Throughout the process, the user must be provided with an interface as intuitive as possible, which provides him with feedback.

SoC-FPGA systems are ideal for implementing such a system. On the one hand, the FPGA is available to handle the acquisition and processing of the signals and, on the other, the microprocessor is ideal to carry the user interface. The classification of the signals and the control of the devices can be handled by either of the two modules. As a criterion for choosing which one to use, evaluating what type of devices will be controlled is proposed. If the purpose of the BCI is to control software (a speller, for example) it can be programmed directly into the microprocessor, together with the signal classifier system. Alternatively, if the idea is to control some external hardware (a wheelchair, for example) it is preferable that the FPGA takes care of both the classifier system and the device driver, to minimize the latency. If it is necessary to control external stimuli to handle the device (visual stimuli, for example), it is preferable to control them by the FPGA, to facilitate their synchronism with the acquired signal.

To guarantee the correct acquisition, storage and processing of the signal there are some basic elements that must be solved within the FPGA. An analog-to-digital converter (ADC) with 12-bit resolution can be used to acquire the signal, as long as an appropriate analog signal conditioning stage is provided. This converter must be managed by a dedicated module in the FPGA. Raw samples should be first saved internally and then exported to some memory shared with the microprocessor, when a certain number of samples is reached. This is done in order for the processor to have access to the raw samples; for graphing, for example, but without having to handle them in real-time.

While there are many possible digital signal processing techniques and methods that can be useful in a BCI system, one of the main ones is the discrete Fourier transform (DFT). This transform provides a way to analyze signals in a time-independent fashion, and the fast Fourier transform (FFT) implementation of it dramatically reduces the number of operations involved on its calculation. Additionally, using this algorithm, correlations and convolutions can be implemented, in general, using fewer operations, which allows the implementation of digital filters in an efficient way [16]. In this scheme, it is the FPGA that must be in charge of calculating the FFT of the signals, saving the results of the processing in some memory shared with the microprocessor. It is desirable that the length of the FFT matches the number of samples that the microprocessor has access to, in order for it to have access to the raw and processed signal in the last time-window at any given moment.

With the acquisition and processing of the signal resolved, it is the microprocessor that must take care of the high-level tasks. The most important one is the user interface, which should be as intuitive and user-friendly as possible. It is desirable that this interface provides the operator with a way to verify the correct acquisition of the signal, by graphing it, for example. In general, these devices carry an operating system, which gives developers the possibility of programming the interface with the high-level design tools they are most familiar with.

2.3. SSVEP-based BCI implementation
2.3.1. Steady state visually evoked potentials

SSEVPs are periodic potentials that appear in a user's EEG record when presented with periodic flashing lights, with frequencies above 6 Hz, and present the same periodicity of the stimulus. Implementing a system that uses these potentials to control some type of device involves measuring the EEG signal, and processing it in its search. There are mainly two ways to use visual stimuli to encode information: phase and frequency encoding. In the first case, the user is presented with multiple flashing stimuli at the same frequency, but out of phase with each other [14]. It is the task of the classification algorithm to detect this delay in order to determine which stimulus the user is looking at. In the second case, the user is presented with visual stimuli blinking at different frequencies [12]. The classification algorithm in this case must determine which is the predominant frequency in the obtained record. Mixed schemes may also be used, with stimuli flashing in both different frequencies and phases [4].

2.3.2. General scheme of the device

In this article a SoC-FPGA was used to implement a SSVEP-based BCI with frequency encoding. The EEG signal is measured using wet electrodes in the user's occipital area. It then goes through a stage of analog signal conditioning, before being digitized by the LTC2308 ADC, available on the de10-nano board. The signal is stored internally and processed using FFT algorithm in the FPGA, and is later transferred to the microprocessor through an integrated bus. The microprocessor is responsible of implementing a speller and a signal graphing tool, using the information provided by the FPGA. At the same time, 5 visual stimuli of different frequencies are generated, which allow the user to control the device. The design of the system was planned in a modular way, seeking that each module serves a specific purpose and is independent of the others. In this way, a widely reusable system is achieved. Figure 1 shows a general diagram of the implemented device.

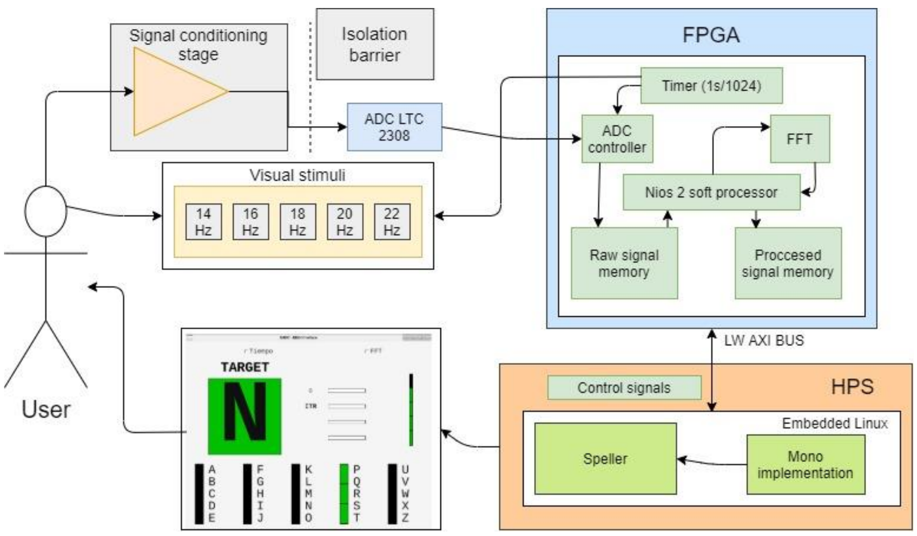


Figure 1. General scheme of the device

2.3.3. Signal conditioning stage and isolation barrier

The EEG signal's amplitude is between 2-100 µV, its frequency ranges from 0.5 Hz to 100 Hz, it's mounted on a DC component that can reach hundreds of mV (electrode potentials) and it's immersed in noise and electromagnetic interference [17]. Once the signal is measured through the wet electrodes an analogic conditioning stage needs to be provided, to enable its acquisition with the ADC integrated in the board (12 bits, successive approximations). For this purpose, a one channel AC coupled differential amplifier was used [18]. This circuit provides an amplification of 5832, and includes a feedback stage through a third electrode, with the purpose of reducing the common mode voltage in the differential pair (usually called a driven right leg circuit, or DRL, in the bibliography [19]). The same amplification stage, with a lower differential mode gain, can be used for obtaining electromyographic (EMG) [20] or electrocardiographic (ECG) [18], [21] signals. The amplifier was complemented with an integrated medical grade isolator ADUM6401, for its power supply, and an optical isolation amplifier based on the IL300 optocoupler, to isolate the signal at its output [22], safeguarding the integrity of the user. An analogic anti-aliasing active filter with cut off frequency of 160 Hz was also added to the set.

2.3.4. Visual stimuli

To generate the visual stimuli by which the user controls the device, 5 matrices of red LED lights, flashing at 14, 16, 18, 20 and 22 Hz, were used. These visual stimuli were suited horizontally, in line with the options available in the user interface, as seen in Figure 2. In order to correctly register the SSVEP phenomenon it's convenient that the periodicity of the stimuli is synchronized with the signal sampling time so the same timer, set at 1/1024 s and generated in the FPGA, was used to control both modules.

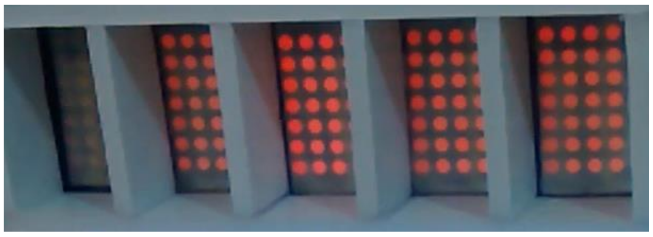


Figure 2. Visual stimuli layout

2.3.5. Low level processing

The acquisition and storage of the signal and the calculation of its FFT were implemented in the FPGA, coded using the Verilog hardware description language (HDL). Figure 3 shows in more detail the different modules that make up the digital design. These are the ADC driver, the visual stimuli driver, the timer, the FFT module, and memories and control signals that interfaces the system with the HPS. In this design a NIOS 2 soft processor was also included. The whole design is synchronized through a 50 MHz clock, available in the de10-nano board.

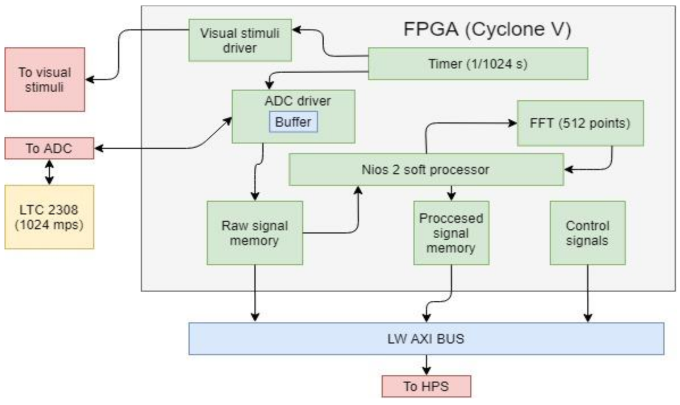


Figure 3. FPGA implementation detail

The visual stimuli are controlled through a module that counts positive edges on the timer's clock, generating square signals of the desired frequencies that are synchronized with the timer. The ADC LTC2308, external to the SoC but included on the board, is driven by SPI protocol through a dedicated module. The timer fixes the sample rate at 1024 samples per second (sps). The samples from the ADC are firstly stored in an internal buffer, with a capacity of 512 samples. Once this buffer is filled the calculation of the sample's 512-point FFT takes place, and both the raw and the processed samples are stored in memories shared with the HPS, which is notified. This process is controlled and monitored by the NIOS 2 processor, which is a "soft" processor, i.e., programmed into the logic cells of the FPGA. The use of this processor was considered vital in the system development and debugging stage, since it provides greater versatility and control over the different stages of the process, and can be programmed easily through a hardware abstraction layer (HAL) provided by Intel Altera. In a future implementation of the system, it could be dispensed with, developing some direct communication between the ADC driver and the FFT module. This would allow a more efficient system, while saving space in the FPGA logic cells. No pre-processing is conducted on the samples provided by the ADC before inputting them to the FFT module, since it was considered unnecessary for the application intended at this stage. In future implementations of the system the data could be pre-processed using FIR [23] or IIR [24] filters, to reduce the off-band noise, and adaptative filters, for example a least-mean square filter [25], to reduce the on-band noise, produced by muscle artifacts. This will surely result in a more robust system.

The module that implements the FFT is made up of two parts. A core provided by [26] is responsible for calculating the 512-point FFT with a serial input/output interface, while two FIFO memories and dedicated logic solve the input/output interface with the NIOS processor. The core receives input samples in 12-bit format, computes the FFT and provides 16-bit outputs, truncating the values if necessary. To help reduce the internal truncation error of the core, while keeping the resource utilization reasonable, all the computations inside of it are carried out in 20-bit words. This includes the twiddle factors necessary for computation, that are stored in look-up tables to avoid having to compute them on the fly. The combination of a 512-point FFT with a sampling frequency of 1024 sps gives the system a frequency resolution of 2 Hz, so that all the frequencies of interest (14, 16, 18, 20 and 22 Hz) correspond to whole beams of the FFT, minimizing the spectral leakage. Once the computation is done the output of the FFT is stored in 32-bit length words, to facilitate its interface with the rest of the system.

In order to implement communication between the modules, two different standard Altera buses were used, as shown in Figure 4. When the ADC driver notifies that the 512 samples are available in the "Raw signal memory", the NIOS 2 processor transfers them to a first in first out (FIFO) memory (FIFO 1 in the figure). These transfers are managed by an Avalon memory mapped interface, which is well suited to implement read and write interfaces for master and slave components such as this. Then the data flows through the FFT. For this, Avalon streaming interfaces, which are ideal for low-latency, unidirectional data transfers, were used. Finally, the processed data is available in another FIFO memory (FIFO 2 in the figure), and the NIOS 2 processor transfers them to the "processed signal memory", using Avalon memory mapped interfaces again. This memory, alongside with the "raw signal memory" is available to the HPS, through the lightweight-AXI bus. Both memories, implemented with on-chip RAM, store data in 32-bit length words. For more information about the Avalon interfaces refer to Altera documentation [27].

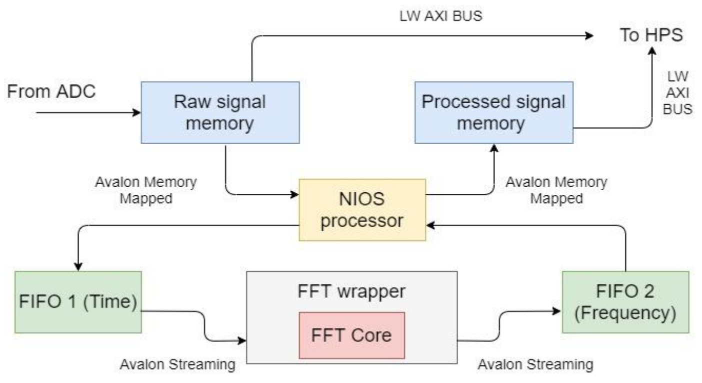


Figure 4. FFT module detail

2.3.6. HPS and FPGA communication

Interfacing the FPGA and the HPS involves selecting one of the three buses integrated in the SoC-FPGA. These are: the FPGA-to-HPS Bridge, the HPS-to-FPGA Bridge and the Lightweight HPS-to-FPGA Bridge. In this design there are no HPS's slaves that need to be controlled from the FPGA, all the transactions that need to take place have a maximum width of 32 bits, and there is no need for a high-bandwidth bus, since all the real time processing takes place in the FPGA fabric. For these reasons the lightweight HPS-to-FPGA Bridge was selected. This bus is driven by a 100 MHz clock, has a fixed data width of 32 bits and exposes an interface that the user can connect to Avalon Memory Mapped slave interfaces. The manufacturer recommends its use for low-bandwidth traffic [27].

2.3.7. High level processing

The high-level processing was coded in the hard processor system (HPS) of the device. This module has direct access to the raw and processed signal in the last window of 512 samples. As seen in Figure 5, information is directly mapped in the virtual memory of the embedded Linux running in the processor, at the character device “dev/mem”. A program running in Linux must access this device and read the information, which is synchronized by the FPGA modules through the control signals. In this scheme the developer can design high level programs using the tools that he or she is more used to. For this article a classifier algorithm and a speller were programmed in C#, executing them through the mono implementation of the .NET framework [28].

In each time-window (512 samples or 0.5 seconds) the classifier algorithm compares the magnitude of the signal’s FFT at the frequencies of interest (14, 16, 18, 20 and 22 Hz), with each other and against the mean, selecting the biggest one. If a frequency is selected in three consecutive time-windows it’s inferred that the user was looking at the visual stimulus related to that frequency. Using this system, a user can select between five different options, without moving any other muscle than the eyes. The criterion of the three consecutive windows was adopted to reduce the probability of false positives, with the downside of reducing the achievable ITR of the system.

By relating each of the visual stimulus to a symbol in the screen a speller was implemented. The system allows the user to select between 5 symbols so, in a first stage, the GUI, which can be seen in Figure 1, asks the user to select between five groups of five letters. Later, in a second stage, the user is asked to select between the five letters of the selected group. Using this scheme, a total amount of 25 letters can be selected, with a minimum time of 3 seconds per selection (1,5 second per command). The “Y” letter was omitted due to its low appearance in the Spanish language. The GUI also provides the user with progress bars, related to the three successive selections, and the possibility of observing the raw and processed signal graphics.

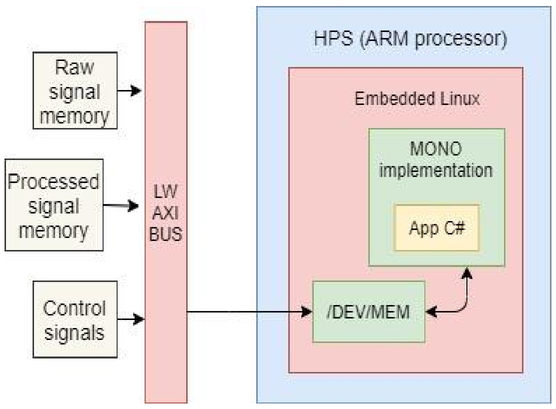


Figure 5. HPS module detail

3. RESULTS AND DISCUSSION

3.1. Usage of FPGA resources

The design was implemented in the de10-nano board. Table 1 summarizes the resources utilized to fit it on the board. This summary shows that the design would even fit in smaller FPGA devices, and that there are plenty of resources left to implement further signal processing elements. Additionally Figure 6 shows the floorplan of the proposed system, obtained from the chip planner perspective on the software provided by Altera, for the Cyclone V 5CSEBA6U23I7 system. This Figure illustrates the area covered by the orchestrated circuit, which is the region covered by square shapes bluer than the others.

Table 1. Resource utilization

	Used	Available	Percentage
ALMS	6157	41910	15%
Memory blocks (BITS)	1433366	5662720	25%
M10K blocks	214	553	38%
DSP blocks	15	112	13%
Registers	10665	83820	12%

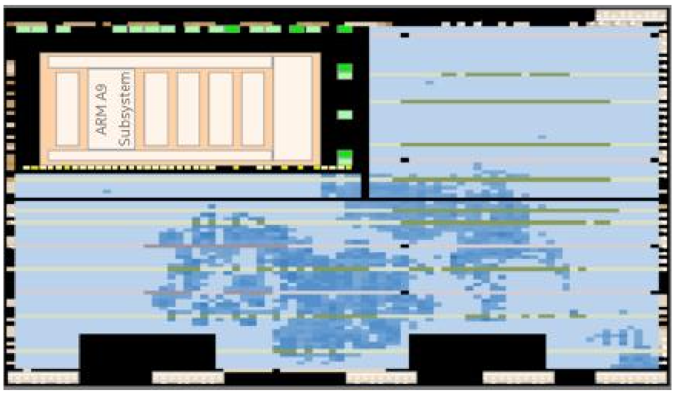


Figure 6. Floorplan of the proposed system

3.2. Register transfer level diagram of the system

Figure 7 shows the register transfer level (RTL) diagram of the proposed system. The block indicated as "soc_system: ADC" is generated by the system integration tool (known as QSYS) of the software provided by Intel Altera, and includes the ADC interface, the FFT core and interconnections, FIFO memories, the Nios 2 processor, the necessary memory for its operation and the HPS interface, as well as some interconnection blocks generated by the tool. The detail of this block is shown in Figure 8.

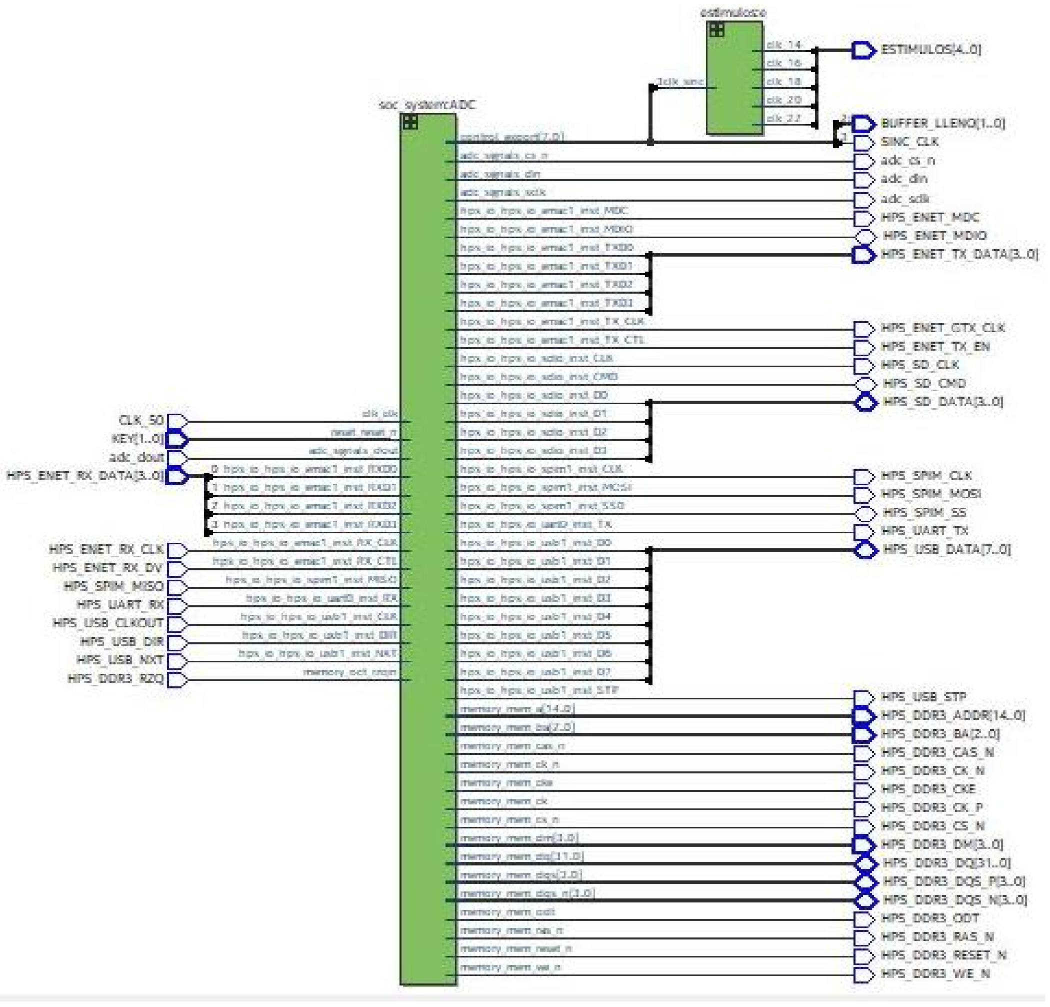


Figure 7. RTL diagram

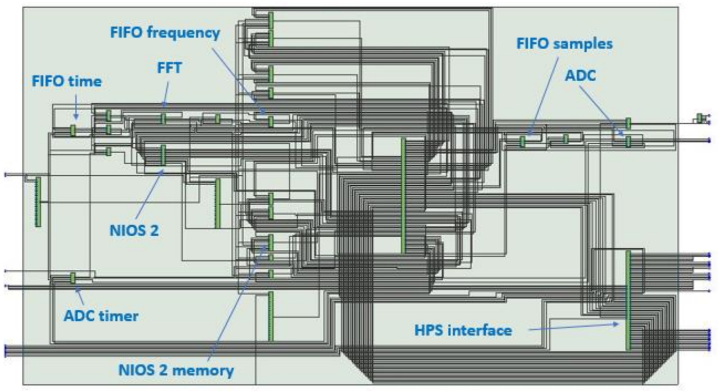


Figure 8. Detail of the "soc_system: ADC" module

3.3. Estimation of the system's latency

After each 0.5 seconds time-window, the calculation of the FFT is conducted and the information is transferred to the HPS module. This results in a latency between the filling of the memories and the moment when the information is available to the user. This latency is made up of the time the NIOS 2 processor takes to transfer the samples from the intermediate FIFO memories (FIFO 1 and FIFO 2 in Figure 4), the time it takes the samples to flow through the FFT, the time that takes the Lightweight HPS to FPGA bridge to conduct the transactions between the modules, and the time it takes the decision algorithm to decide between the different stimuli.

The NIOS 2 processor is driven by the same 50 MHz clock than the whole system, but the clock that drives the hardware abstraction layer (HAL) from which the instructions are conducted is lower [29]. This clock was estimated in 11 MHz, using an oscilloscope and a simple software routine. The FFT module outputs one processed sample per clock after a delay of 1105 clocks, and it is driven by a 50 MHz clock. The decision algorithm time was measured using the C# Diagnostics library, resulting in 14 μs. The transactions between the HPS and the FPGA in Cyclone V devices has been the subject of the study [30]. Alongside with this paper an Excel Table is provided that allows the users to search the measured transfer rate for several modes of operation. For transferring 4096 Bytes (512, 32 bits raw and processed samples) using the Lightweight HPS-to-FPGA bus, the Angstrom Linux operating system and a 50 MHz clock for the FPGA a 20,08 MB/s transfer rate is reported. So, the estimated latency (L) is given by (1), (2) and (3):

$$\mathrm{L} = \mathrm{Time}_{\mathrm{nios}} + \mathrm{Time}_{\mathrm{FFT}} + \mathrm{Time}_{\mathrm{LW}_{\mathrm{bus}}} + \mathrm{Time}_{\mathrm{decision}} \tag{1}$$

$$\mathrm{L} = 2 * 512 * \frac{1}{11\ \mathrm{MHz}} + (1105 + 512) * \frac{1}{50\mathrm{MHZ}} + \frac{2*2048}{20.08\mathrm{MB/s}} + 14\mu\mathrm{s} \tag{2}$$

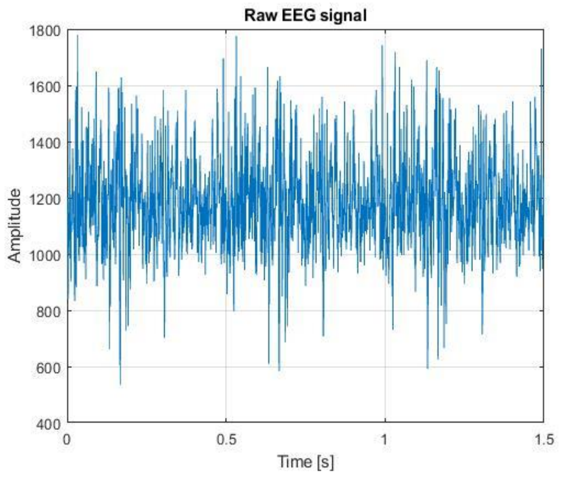
$$\mathrm{L} = 93\mu\mathrm{s} + 32\mu\mathrm{s} + 204\mu\mathrm{s} + 14\mu\mathrm{s} = 343\mu\mathrm{s} \tag{3}$$

This calculation is an estimate but gives an idea of the expected latency of the system. As it can be appreciated the components that mostly contribute with this time delay are the NIOS 2 processor and the Lightweight bus. The first one can be dispensed with in future implementations, while the second one can be replaced with the HPS-to-FPGA bus which support 128-bit transactions, achieving a 35 MB/s transfer rate in these same conditions [30]. A higher frequency in the FPGA may also be used. Without any changes in the design a quick evaluation with the timing analyzer provided by Intel Altera shows that a 72 MHz clock can be achieved. This maximum frequency is computed as the maximum possible at which the transfers between registers are conducted in a single clock cycle. It may be further increased taking special care at some register paths, and evaluating in which paths this condition may be relaxed. With that being said, the authors consider that a latency on the order of the hundreds of microseconds (343 μs in our calculation) is acceptable, as it is almost indistinguishable for the user.

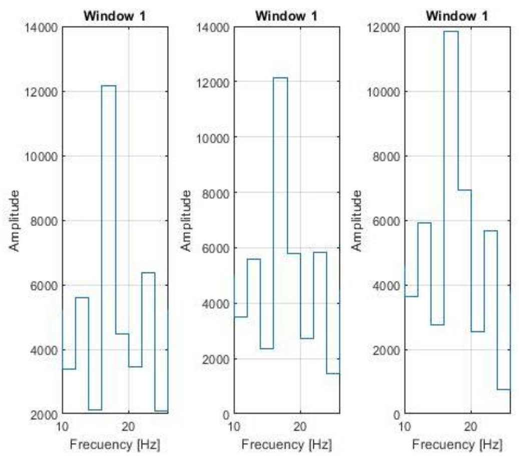
3.4. Speller test

In order to verify the system's overall performance, a simple high-level experiment was conducted. Two users, one with no previous experience in handling BCIs (user A) and one experienced (user B), were asked to select letters randomly. In the best cases the command selection time for user A was about 5 seconds but, in some cases, it took him more than 100 seconds to select a command. On the other hand, user B was able to execute a command typically between 5 and 10 seconds.

These results, that are shown for demonstrative purposes, are comparable with those presented in [14], where the typical user selection time was between 5 and 8 seconds. As an example, Figure 9 shows the EEG record of user A, obtained in 3 consecutive time-windows, corresponding to a successful selection of the option related to a visual stimulus of 16 Hz. The first image corresponds to the raw EEG signal, while the second shows its FFT transform in each time-window.



(a)



(b)

Figure 9. EEG signal in three consecutive time-windows: (a) Raw signal (b) Processed signal

4. CONCLUSIONS

SoC FPGA systems present very attractive characteristics for the development of systems where real-time processing is key. Using these devices, dedicated logic in the FPGA can be used to solve tasks that demand low latency, leaving the processor dedicated to perform high-level tasks, such as the user interface. Throughout this article it has been discussed how a system of these characteristics can be used to develop a platform for the acquisition and processing of electroencephalographic signals, specifying how the tasks can be distributed in order to achieve a highly efficient BCI. Finally, the implementation of an SSVEP-based BCI system, developed entirely on a de10-nano SoC provided by Altera, has been shown. The time delay between the user's selection and the system response has been estimated in 343 µs, and a first experimental test, that allowed to verify the operation of the complete system, was carried out. A detailed characterization of the system performance and the proposal of strategies to improve it remain to be conducted. Although the design has been oriented to obtain and process SSVEPs with frequency encoding, the modularity of the design allows most of the developed tools to be reused in the implementation of different types of BCIs in the future.

REFERENCES

[1] A. Patil and A. A. Shirolkar, "A review on system-on-chip (SoC) designs for real.time industrial application," *International Journal of Tren in Scientific Research and Development (IJTSRD)*, vol 2, no. 1, pp.1534-1537, 2017, doi: 10.31142/ijtsrd7077.
[2] E. Monmasson and M. N. Cirstea, "FPGA design methodology for industrial control systems—A review," in *IEEE Transactions on Industrial Electronics*, vol. 54, no. 4, pp. 1824-1842, Aug. 2007, doi: 10.1109/TIE.2007.898281.
[3] J. R Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller and T. M. Vaughan, "Brain-computer interfaces for communication and control," *Clinical Neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002, doi: 10.1016/S1388-2457(02)00057-3.
[4] X. Chen, Y. Wang, M. Nakanishi, X. Gao, T.-P. Jung, and S. Gao, "High speed spelling with a brain–computer interface," *Proceedings of the National Academy of Sciences*, vol. 112, no. 44, 2015, doi: 10.1073/pnas.1508080112.
[5] A. Chabuda, P. Durka and J. Żygierewicz, "High frequency SSVEP-BCI with hardware stimuli control and phase-synchronized comb filter," in *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 2, pp. 344-352, Feb. 2018, doi: 10.1109/TNSRE.2017.2734164.

[6] J. Long, Y. Li, H. Wang, T. Yu, J. Pan and F. Li, "A hybrid brain computer interface to control the direction and speed of a simulated or real wheelchair," in *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 20, no. 5, pp. 720-729, Sept. 2012, doi: 10.1109/TNSRE.2012.2197221.

[7] M. Pham *et al*., "An auditory brain-computer interface based on the self-regulation of slow cortical potentials," *Neurorehabilitation and Neural Repair*, vol. 19, no. 3, pp. 206-218, 2005, doi: 10.1177/1545968305277628.

[8] H.-J. Hwang, K. Kwon and C.-H. Im, "Neurofeedback-based motor imagery training for brain–computer interface (BCI)," *Journal of Neuroscience Methods*, vol. 179, no. 1, pp. 150-156, 2009, doi: 10.1016/j.jneumeth.2009.01.015.

[9] Y. Li *et al*., "P300 based BCI messenger," *2009 ICME International Conference on Complex Medical Engineering*, 2009, pp. 1-5, doi: 10.1109/ICCME.2009.4906636.

[10] S. M. Lai, Z. Zhang, Y. S. Hung, Z. Niu and C. Chang, "A chromatic transient visual evoked potential based encoding/decoding approach for brain–computer interface," in *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 1, no. 4, pp. 578-589, Dec. 2011, doi: 10.1109/JETCAS.2011.2178734.

[11] R. Srinivasan, F. A. Bibi and P. L. Nunez, "Steady-state visual evoked potentials: distributed local sources and wave-like dynamics are sensitive to flicker frequency," *Brain Topography*, vol. 18, pp 167-187, 2006, doi: 10.1007/s10548-006-0267-4.

[12] P. A. García, E. M. Spinelli, G. M. Toccaceli, "An embedded system for evoked biopotential acquisition and processing," *International Journal of Embedded Systems (IJES)*, vol. 6, no. 1, pp. 86-93, 2014, doi: 10.1504/IJES.2014.060920.

[13] Xiaorong Gao, Dingfeng Xu, Ming Cheng and Shangkai Gao, "A BCI-based environmental controller for the motion-disabled," in *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 11, no. 2, pp. 137-140, June 2003, doi: 10.1109/TNSRE.2003.814449.

[14] J.-S. Lin and W.-C. Wu, "An FPGA-based BCI system with SSVEP and phased coding techniques," *Journal of Technology*, vol. 33, no. 1, pp. 53-62, 2018.

[15] B.-S. Lin, B.-S. Lin, T.-H. Yen, C.-C. Hsu and Y.-C. Wang, "Design of wearable headset with steady state visually evoked potential-based brain computer interface," *Micromachines*, vol. 10, no. 10, p. 681, 2019, doi: 10.3390/mi10100681.

[16] B. Porat, "The fast Fourier transform," in *A course in Digital Signal Processing*, 2th ed. United States: John Wiley and Sons Inc, 1997, ch. 5, pp 133-163.

[17] J. D. Bronzino, "The biomedical engineering handbook" USA:CRC PressI Llc. Trinity College Hartford, Connecticut, 2000, vol. I, sc. 6.

[18] E. M. Spinelli, N. H. Martinez and M. A. Mayosky, "A single supply biopotential amplifier," *Medical Engineering and Physics*, vol. 23, no. 3, pp. 235-238, 2001, doi: 10.1016/S1350-4533(01)00040-6.

[19] B. B. Winter and J. G. Webster, "Driven-right-leg circuit design," in *IEEE Transactions on Biomedical Engineering*, vol. BME-30, no. 1, pp. 62-66, Jan. 1983, doi: 10.1109/TBME.1983.325168.

[20] H. İ. Çakar, S. Kara and O. Toker, "Design of a portable electromyography device for analysis of back herniation," *2010 15th National Biomedical Engineering Meeting*, 2010, pp. 1-3, doi: 10.1109/BIYOMUT.2010.5479839.

[21] E. M. Spinelli, R. Pallas-Areny and M. A. Mayosky, "AC-coupled front-end for biopotential measurements," in *IEEE Transactions on Biomedical Engineering,* vol. 50, no. 3, pp. 391-395, March 2003, doi: 10.1109/TBME.2003.808826.

[22] D. Görk and A. M. Kruck, *Designing Linear Amplifiers Using the IL300 Optocoupler*, Vishay Semiconductors Application note 50, Document Number: 83708, 2017.

[23] A. Mahabub, "Design and implementation of cost-effective simple FIR filter for EEG signal on FPGA," *World Scientific News*, vol. 125, pp. 1-17, 2019.

[24] A. Mahabub, "Design and implementation of cost-effective IIR filter for EEG signal on FPGA," *Australian Journal of Electrical and ElectronicsEngineering*, vol. 17, no. 2. Pp. 83-91, 2020, doi: 10.1080/1448837X.2020.1771662.

[25] P. V. Dutande, S. L. Nalbalwar and S. V. Khobragade, "FPGA implementation of filters for removing muscle artefacts from EEG signals," *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2018, pp. 728-732, doi: 10.1109/ICCONS.2018.8662998.

[26] D. Gisselquist, "An open source pipelined FFT generator," Gisselquist Technology's ZipCPU website, 2018. [Online]. Available: https://zipcpu.com/dsp/2018/10/02/fft.html. Accessed: Aug. 2021.

[27] Inter Corporation, "Avalon® Interface Specifications," 2020. [Online]. Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/manual/mnl_avalon_spec.pdf. Accessed: Aug. 2021.

[28] Mono project, 2020. [Online]. Available: https://www.mono-project.com/. Accessed: Aug. 2021.

[29] Intel Corporation, "Nios® II Software Developer Handbook," 2020, ch. 7, pp. 156-206. [Online]. Available: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/nios2/n2sw_nii5v2gen2.pdf. Accessed: Aug. 2021.

[30] R. F. Molanes, J. J. Rodríguez-Andina and J. Fariña, "Performance Characterization and Design Guidelines for Efficient Processor–FPGA Communication in Cyclone V FPSoCs," in *IEEE Transactions on Industrial Electronics*, vol. 65, no. 5, pp. 4368-4377, May 2018, doi: 10.1109/TIE.2017.2766581.

BIOGRAPHIES OF AUTHORS

Matías Javier Oliva was born in Rio Gallegos, Santa Cruz, Argentina, in 1993. He received the Engineer in electronics degree from La Plata National University (UNLP) in 2018. Since then, he has been working on digital instrumentation and biopotentials in the Biomedical Industrial and Scientific Instrumentation group (GIBIC), dependent on the Research Institute in Electronics, Control and Signal Processing (LEICI) in UNLP. He is also currently an Assistant Professor in the Department of basic sciences, in UNLP. His current research interests are System on Chip devices, digital design, brain computer interfaces and magnetic tomography

Pablo Andrés García was born in Azul, Argentina, in 1976. He received the degrees of Engineer in electronics, Master in Engineering and Doctor in engineering at La Plata National University (UNLP), in 2002, 2008 and 2019 respectively. He is currently a Full Professor in the department, of Electrotechnics of the Faculty of Engineering in the, UNLP and integrates the Instrumentation group, Biomedical, Industrial and Scientific (GIBIC), dependent on the Research Institute in, Electronics, Control and Signal Processing, (LEICI) of the UNLP.

Enrique Mario Spinelli was born in Balcarce, Argentina. He received the degrees of Engineer in, Electronics, Magister and Doctor of Engineering in La Plata National University (UNLP). He is currently a Professor at the Faculty of, Engineering at UNLP, Researcher at CONICET, and Director of the Instrumentation Group, Biomedical, Industrial and Scientific (GIBIC), dependent on the Research Institute in, Electronics, Control and Signal Processing, (LEICI) UNLP-CONICET

Alejandro Luis Veiga received the degrees of Engineer in Electronics, Magister and Doctor of Engineering at La Plata National University (UNLP) in 1993, 1999 and 2008 respectively. He is currently a Professor at the Faculty of, Engineering at UNLP, Researcher at CONICET, and member of the Instrumentation Group, Biomedical, Industrial and Scientific (GIBIC), dependent on the Research Institute in, Electronics, Control and Signal Processing, (LEICI) UNLP-CONICET. His primary research field is Scientific Instrumentation.