Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2024,11(7):1-11

Research Article ISSN: 2394-658X

Blind separation of three signals using Matlab/Simulink software and Arduino Due board

Nguetsop Hermann Beaumont*, Gamom Roland, Jean Mbihi

Research Laboratory of Computer Science Engineering and Automation, ENSET, University of Douala, Cameroon

*corresponding author: beaumont.nguetsop@yahoo.com

ABSTRACT

This work describes the simulation of a blind source separation algorithm for three sound signals in the Matlab/Simulink software and its implementation in the Arduino Due card defined as low-cost hardware. For the sake of clarity, reference signals generated by three generators (GBF) were subsequently used. The algorithm has been modified to match the inputs and outputs of the Arduino board in Real Time. This work proved that our theoretical algorithm could achieve the same result by performing the separation on the Arduino Due to validate its real-time performance in the practical world. It would then be integrated into the FECG measurement instruments in which three signals must be separated.

Key words: blind source separation, BSS, Arduino Due, Matlab/Simulink, Real Time. FECG, MECG, RCF, SIR, SOBI

1. INTRODUCTION

Continuously monitoring the parameters of fetal activity during pregnancy has the immediate benefit of early detection of pathologies in order to allow the intervention of the gynecologist before the appearance of irreversible changes. Referring to the report of the Canadian Society of Obstetrics and Gynecology [1] and that of the National Multisectoral Program to Combat Maternal, Neonatal and Infant-Juvenile Mortality in Cameroon 2014/2020 [2], we note that monitoring more Frequent heart rate, simultaneously with other parameters, allows reduction of perinatal morbidity and mortality up to 7.7‰. In this respect, Doppler ultrasound has shown its advantages in assessing the state of fetal wellbeing [3]. However, this examination requires the presence of the patient and many health centers do not have this. The development of on-board technologies has enabled the birth of numerous portable equipment for measuring FHR, but the only drawback remains the high cost. In the design of a new on-board device for permanent monitoring of the FHR, one of the difficulties encountered is that the FHR signal from the sensors is a mixture of the maternal ECG signal; Fetal FECG; and a set of noises (influenced by the size of the heart which varies with gestational age [4]), the electrical activity of the uterus (more active at the end of pregnancy), other traffic noises blood, as well as by the heart-sensor distance [5]. It is therefore necessary to extract (separate or dislodge) the fetal cardiac signal from this composite signal. Blind separation of sources is a generalization of deinterference (action of reducing noise) of a signal drowned in noise when we have several sensors recording in different ways the same signals called sources) [6]. This method offers a definitive solution to the problems of FECG and consists of estimating a group of signals from unknown sources originating from their observed mixtures when there is very little information available on the mixture model, without additional assumptions [7]. Other works presenting the solution of other similar biological problems having used source separation, particularly in the processing of responses emitted by human muscles subjected to different types of excitations, have since been published. C.Jutten and J.Hérault [8] laid the foundations of the first mathematical equations of the source separation problem and an algorithm to arrive at a solution [9]. Since then, several other works in blind source separation (BSS) have been published using new, more efficient algorithms implemented in real time in signal processing processors, FPGA boards [12] [13], and Arduino Due [15] [21] with two separate signals and using a new ICA algorithm based on the rotation of the Jacobean matrix and the generator blocks of the Matlab system.

None of this work has addressed real-time source separation using three signals and the Arduino Due board as free and inexpensive hardware. In this work, we present a significant modification of the SOBI algorithm allowing the separation of three different audio signals which will be modeled in the Matlab/Simulink software and it will be implemented in the Arduino Due card to test the performance of the latter in real time.

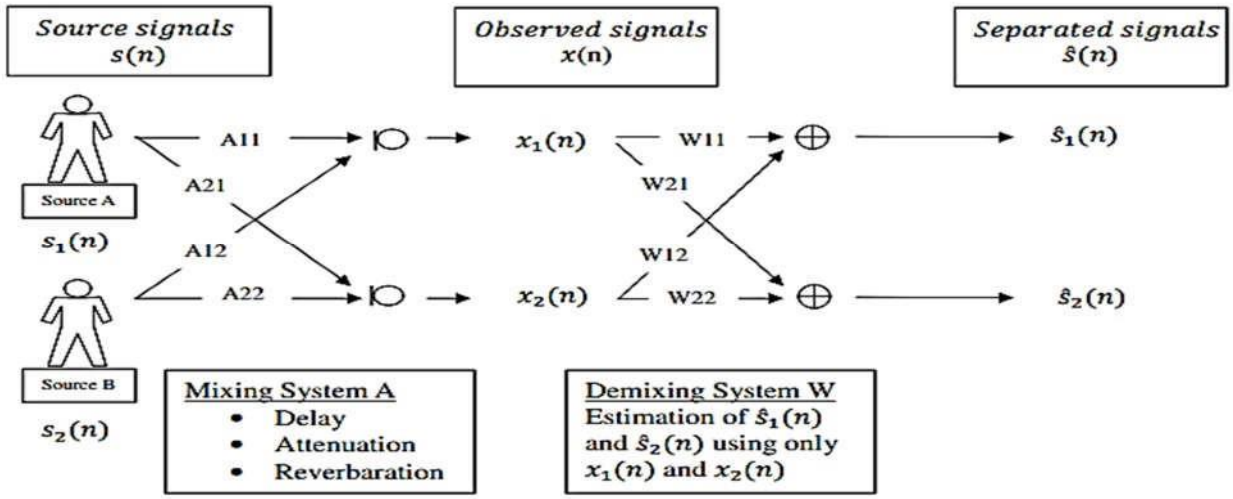

Figure 1: Block diagram of the general principle of blind separation of two sources

2. TOOLS AND METHOD

A. principle of blind source separation

If recording an electrocardiogram on an adult no longer presents too many difficulties, that of a fetus remains a much more complex task. Indeed, the fetal heart rate alone is not sufficient to diagnose possible pathologies: the shape of the heart wave contains important information. But, being in the woman's womb, several natural barriers would have to be crossed (mother's skin, placenta, child's skin) before obtaining the useful signal. For this, a network of sensors placed in different locations in the mother's womb is necessary, because they will be able to continuously record this packet of signals. This signal mixture is unfortunately unusable because it is riddled with several noises which have the following sources:

[1]. The respiration and blood circulation of the mother and that of the fetus.
[2]. The mother's heartbeat (MECG).
[3]. Fetal heartbeat (FECG).
[4]. Noise from the power supply linked to active sensors.
[5]. Sounds from the mother's digestive system or electromyograms (noted EMG).

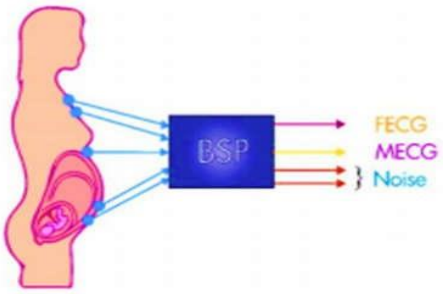

Figure 2: Origin of mixed signals

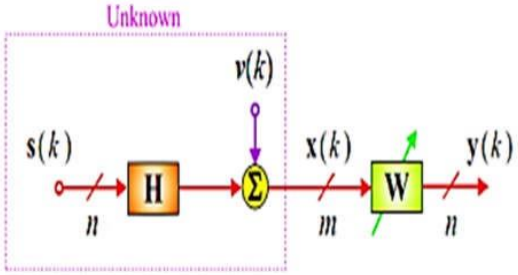

Figure 3: Principle of source separation

Blind source separation (BSS) is based on estimating signals from unknown sources from their observed mixtures when there is very little information available about the mixture model [16]. In our case, the sources are formed by a network of sensors which continuously record several signals of the same nature at different amplitudes: The fetal heart signal (FECG), the mother's heart signal (MECG) and other noises (noise).

The principle of BSS can be represented in Figure 3, it involves finding in the ideal case the matrix W of size n x m which provides the output vector:

$$y(k)=W\,x(k)=W\,H\,s(k)\simeq s(k) \quad (1)$$

The estimated sources are obtained from the vector and their associated projections for the different sensors are determined from the estimated mixing table:

$$H=W^{-1} \quad (2)$$

The cocktail party problem is a classic example of blind source separation (BSS), the separation of a set of observations into constituent (statistically independent) underlying source signals. The cocktail party problem is illustrated in Fig. 4. If each of the k voices you can hear in a market is recorded by M microphones, the recordings will be a matrix composed of a set of M vectors, each of which is a (weighted) linear superposition of the k voices. In the case of three sources; the mixed signal can have this expression.

$$\mathbf{X} = \begin{bmatrix} A1 & A4 & A7 \\ A2 & A5 & A8 \\ A3 & A6 & A9 \end{bmatrix} \begin{bmatrix} S1 \\ S2 \\ S3 \end{bmatrix} = \mathbf{AS} \quad (3)$$

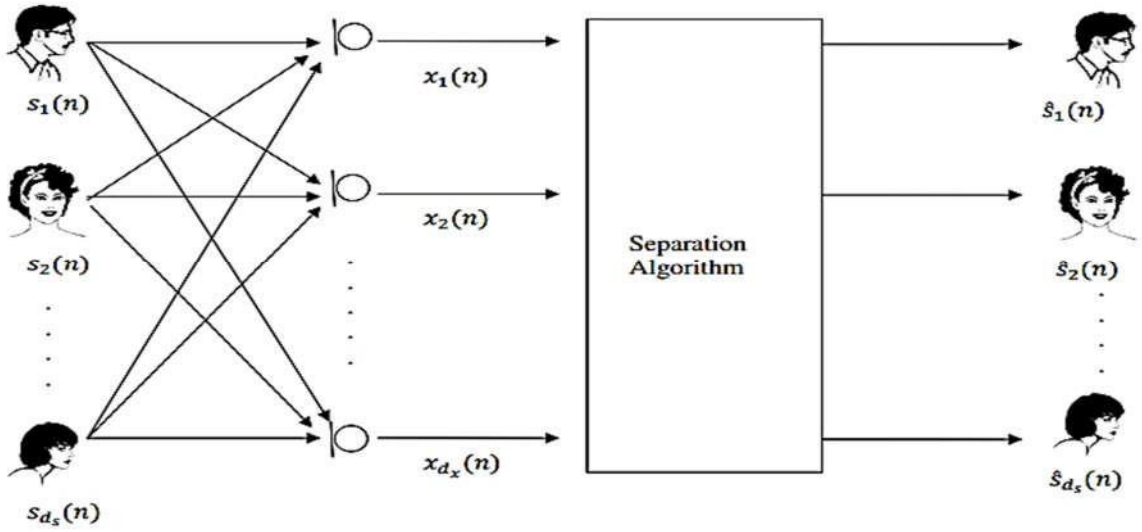

Figure 4: Illustration of the cocktail party problem

For a model without noise and X = A * S + N for a model with noise. With the following elements:

S(t) = [s1(t), s2(t),...,sN(t)]T is the vector of N unknown sources at time t.
X(t) = [x1(t), x2(t),...,xM(t)]T is the vector of the M signals observed at time t.
N(t) = [N1(t), N2(t),...,NM(t)]T is the vector of N noises observed at time t.

A Є R M x N is the unknown mixture matrix which is assumed to be of full rank and invertible. The problem can be divided into three cases depending on the number of sources N and the number of sensors M:

[1]. Overdetermined if M > N,
[2]. Determined if M = N,
[3]. Underdetermined if M < N.

n this study case, the separation is blind overdetermined (where we must estimate the mixing matrix and the sources) and there exist several pairs of solutions (A, S). The ACI structure is made up of four levels:

[1]. The first step consists of whitening the input vector S and reducing its dimension therefore transforming it into a new form:

$$X=WS \quad (4)$$

Such that W is the whitening matrix. The objective of laundering is to normalize second-order statistics.

[2]. The second step is to do the separation via the separation matrix B :

$$Y = BTX \quad (5)$$

The elements of Y are the extracted independent components.

[3]. The third step is to estimate the basis vectors of the ICA.

[4]. The last one is Dimension Reduction: A common preprocessing technique for multidimensional data is to reduce its dimension, i.e., the number of observations in the source separation application. This allows us to eliminate redundancy in observations, so that the computational cost can be alleviated. This preprocessing has great advantages, especially for overdetermined flash mixtures when the number of sources is known. To do this, principal component analysis (PCA) is often used. One of the advantages of this preprocessing is noise reduction.

It was proven in [15] that the effectiveness of the BSS algorithm is measured by observing the value of the separability index SI, through the following equation:

$$SI = \frac{1}{n(n-1)} \sum_{i=1}^{n} \left\{ \left(\sum_{k=1}^{n} \frac{|g_{ik}|}{max_j |g_{ij}|} - 1 \right) + \left(\sum_{k=1}^{n} \frac{|g_{ki}|}{max_j |g_{ji}|} - 1 \right) \right\} \quad (6)$$

The elements (i, j) of the global matrix of the system G=WH (7); and maxj (gij) the maximum value of the elements of the ith row vector of G. In the same way maxj (gij) has the maximum value of the jth theme of the line vector G. When the maximum separation is reached, the separability index is zero. In reality, the SI value to the nearest hundredth 10-2 indicates relatively high performance [15].

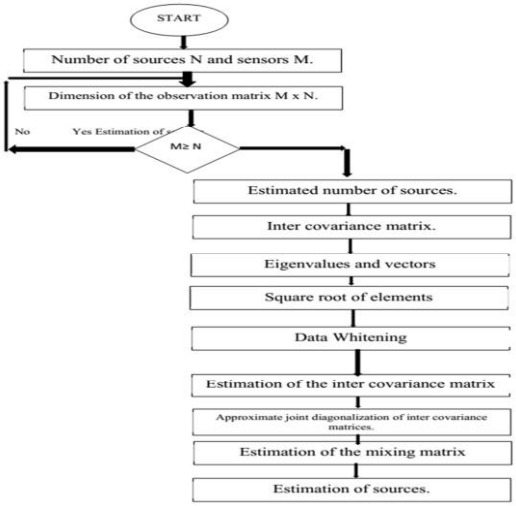

Figure 5: BSS operating flowchart.

B. The Arduino Due board

The Arduino system is an open-source programmed electronics platform which is based on a simple microcontroller board (from the AVR family), and software, a true integrated development environment, for writing, compiling and transferring the program to the map [15].

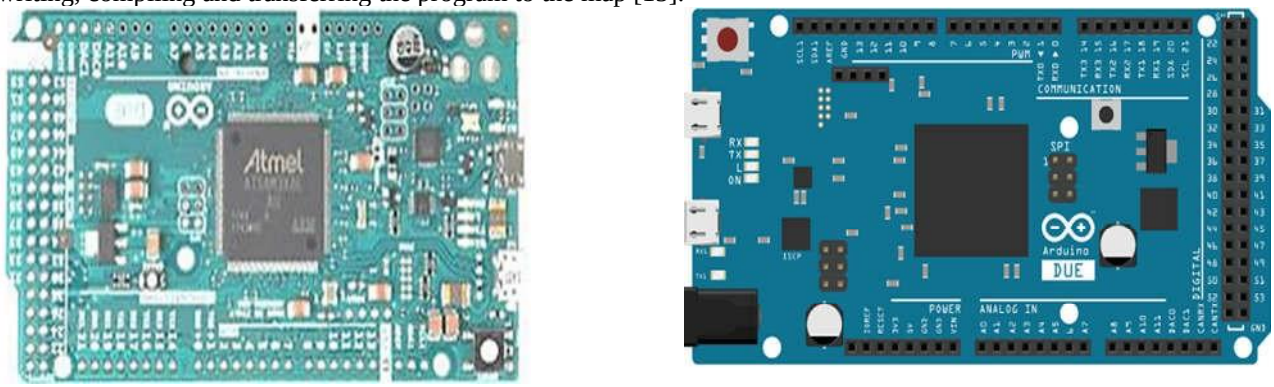

Figure 6: Presentation of the Arduino Due board [15] [19].

Table 1: Features of the Arduino Due board

Microcontroller	**AT91SAM3X8E**
Clock Speed	84 Mhz
Flash Memory	512 KB
Digital I/O Pins	54
SRAM	96 KB
Analog Output Pins	2
Analog Input Pins	12
Operating Voltage	3.3 V
Input Voltage (limits)	6 -16 V
Input Voltage (recommended)	7 12 V
DC Current for 5v Pin	800 mA
DC Current for 3.3V Pin	800 mA
DC output Current	130 mA

C. Proposed work

The objective of this work is to implement with Matlab/Simulink software a modified SOBI source separation algorithm using three audio signals. The results of the software simulation must be compared to real-time results in an Arduino board and visualized via an oscilloscope. For this, we used Simulink blocks and Matlab functions to create our source separation system. The block diagram is shown in the following figure.

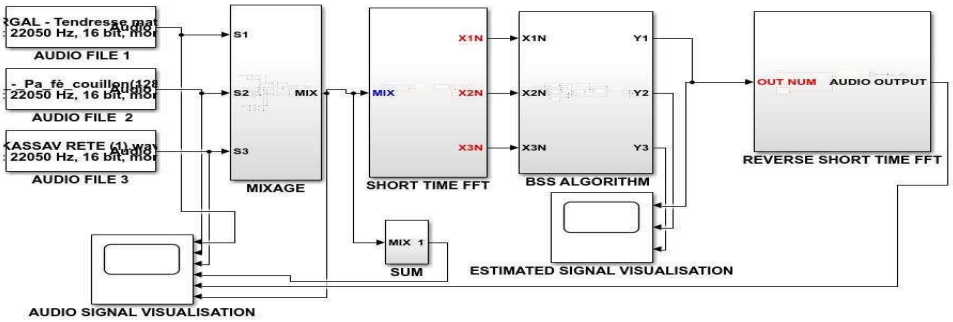

Figure 7: Block diagram of our device on Simulink.

[1]. file: playback of wav files.
[2]. Mixing block: mixes the S1, S2 and S3 signals.
[3]. Short Time FFT: Fast Fourier transform of audio signals. X1; X2 and X3.
[4]. BSS Algorithm: Contains the essentials of the SOBI algorithm.
[5]. Reverse Short Time FFT: Converts digital signal to analog audio signal

D. Audio signal settings.

Table 2: Characteristic tables of the audio files used

TITLE	AUDIO FILE 1	AUDIO FILE 2	AUDIO FILE 3
CHANNELS	Mono	Mono	Mono
SAMPLING RATE	22050	22050	22050
BITRATE (kbits/sec)	16	16	16
SIZE (Mo)	1.12	1.56	1.26
TIME (secs)	30	30	30

The simulation is planned for 30 seconds of operation of each audio file; but we will use a maximum of 4 seconds for visualization in Simulink oscilloscopes.

3. RESULT AND DISCUSSION

A. Simulation results

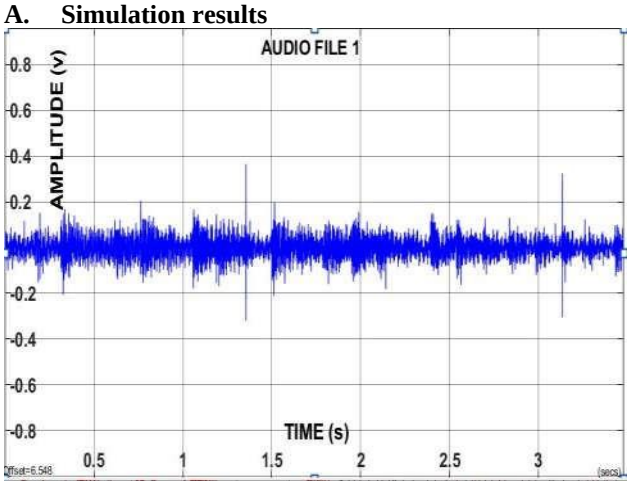

Figure 8: Source signal S1

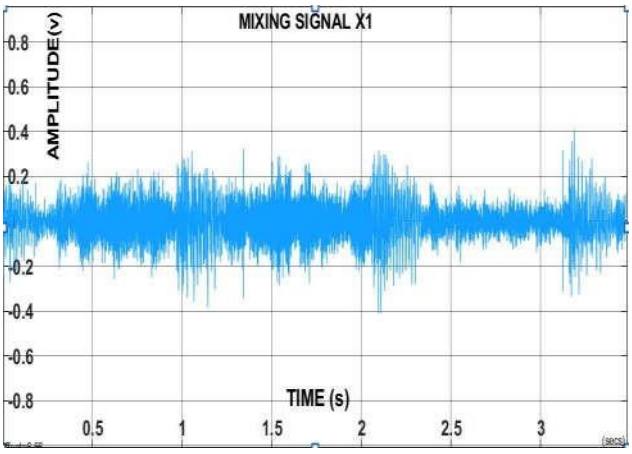

Figure 9: Mixed signal X1

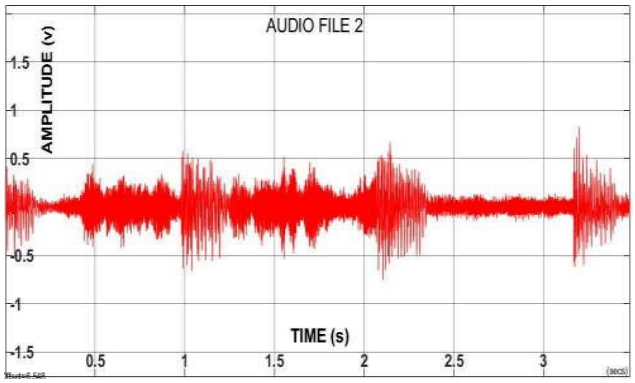

Figure 10: Source signal S2

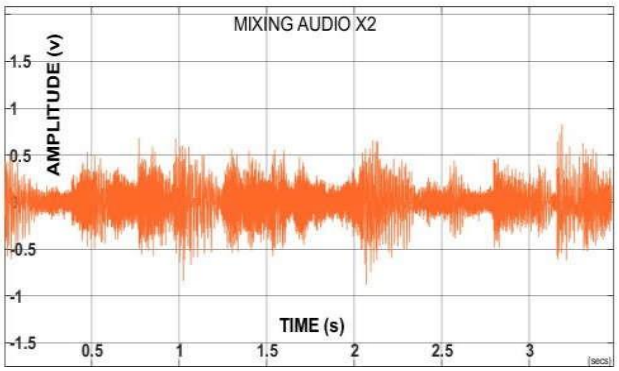

Figure 11: Mixed signal X2

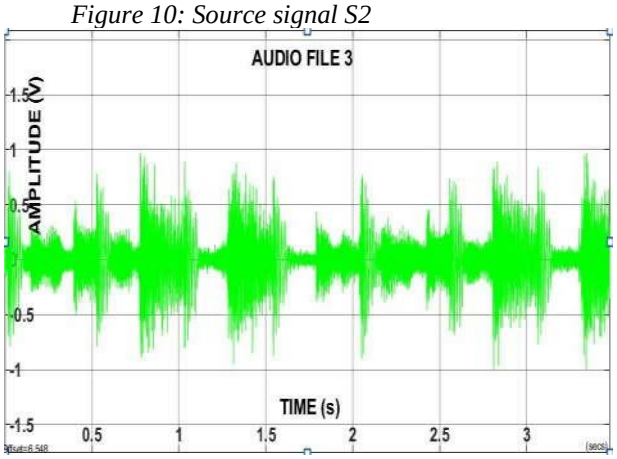

Figure 12: Source signal S3

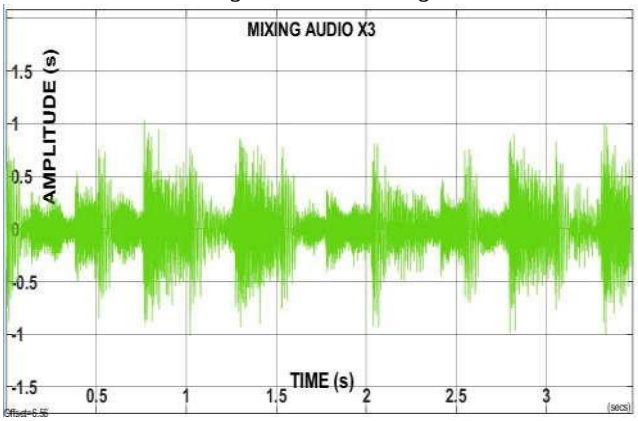

Figure 13: Mixed signal X3

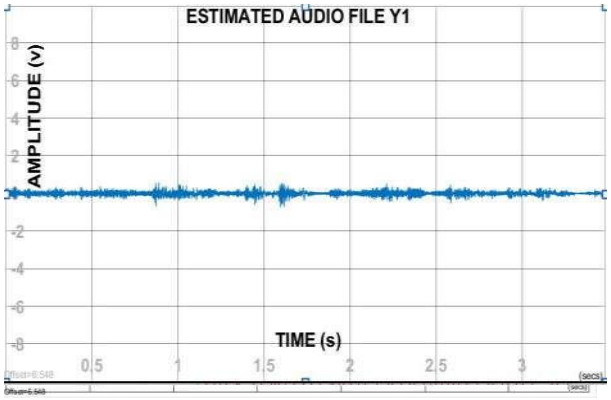

Figure 14 : Estimated signal Y1

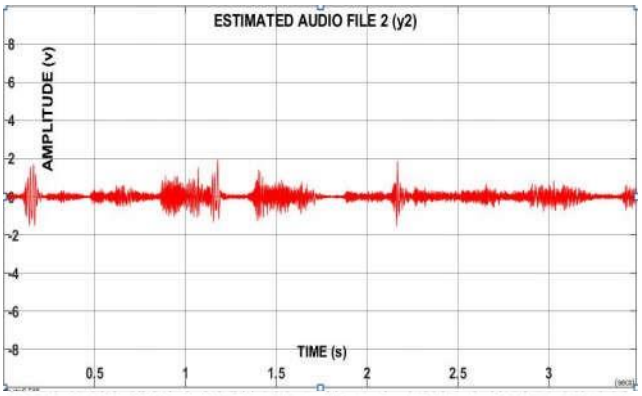

Figure 15 : Estimated signal Y2

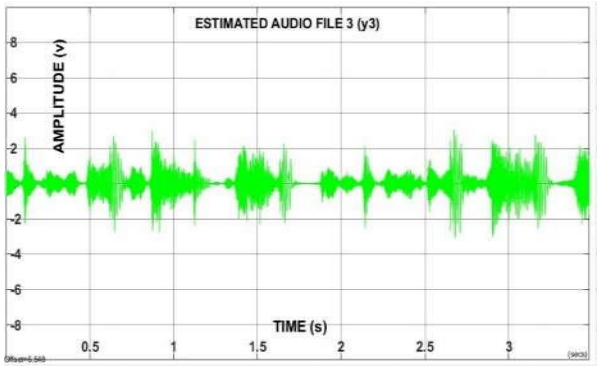

Figure 16 : Estimated signal Y3

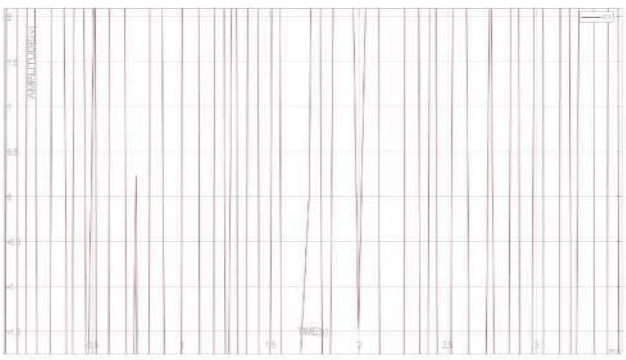

Figure 17 : Sum signal

B. Data the implementation in the Arduino board.

We used the following software elements:

[1]. Windows 10 operating system.

[2]. Matlab/Simulink 2023b software.

And the hardware elements: For clear observation of the operation of the system, we will use reference signals other than the audio signal.

[3]. low frequency generators (GBF) to generate 03 signals

A. Square signal with frequency 693Hz ; and amplitude 5v.

B. Sinusoidal signal with frequency 50Hz and amplitude 10v.

C. A triangular signal with a frequency of 118Hz and an amplitude of 667mv.

[4]. 01 bi-curve digital oscilloscope (UNIT-T/UTD2025CL) of 25MHz. An Arduino Due board.

To transform our simulation block diagram to use Arduino I/O in Matlab, the Simulink Support Package Library for Arduino hardware must be installed. It can be obtained and installed by clicking on the (Add-Ons) tab, then (Get Hardware Support Package) of the Matlab/Simulink software as shown in Figure 20. Simulink must be configured and used with Arduino Due. In the simulation options, external mode is enabled to work in real time with external hardware. When execution is requested, RTW converts the Simulink block diagram into C/C++ code, in order to compile and download the code into the microcontroller. The code is executed in real time with the sampling time chosen in the blocks of Figure 18; and the selected signals are sent to Simulink via USB interface.

igure 19 shows the flowchart of the Real Time implementation [19]. In this part, our BSS algorithm was implemented in an Arduino Due card and the tests were gradually carried out. This resulted in a set of curves verifying the effective functioning of our algorithm.

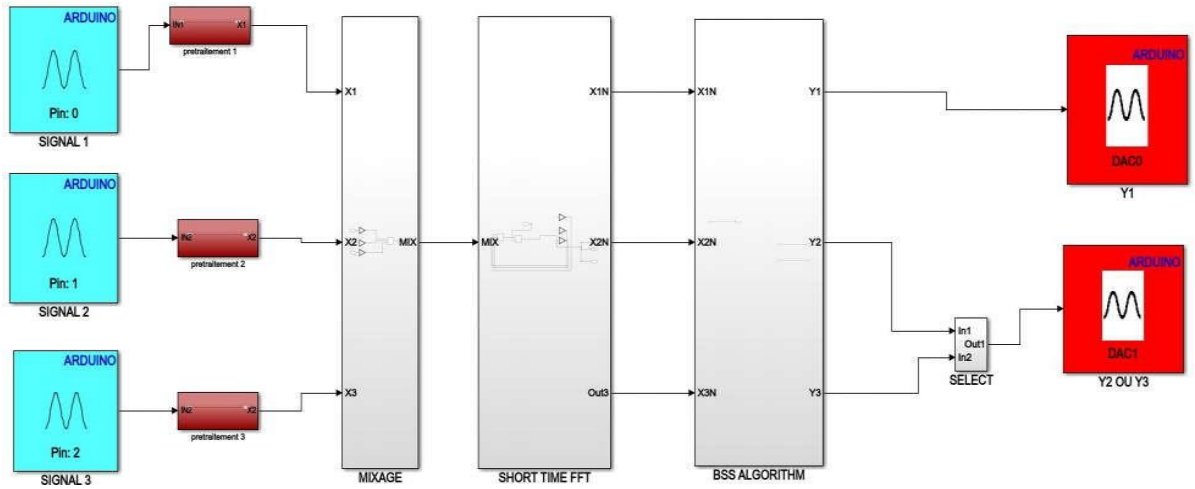

Figure 18 : Block diagram of our model formatted with the inputs and outputs of our Arduino Due board.

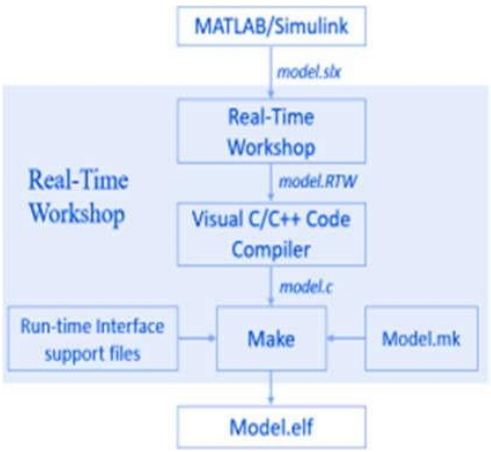

Figure 19 : Implementation flow chart

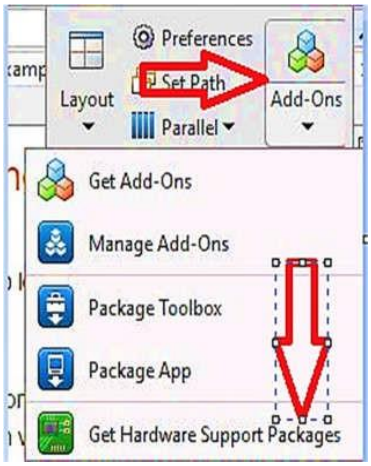

Figure 20 : Installing Arduino packages

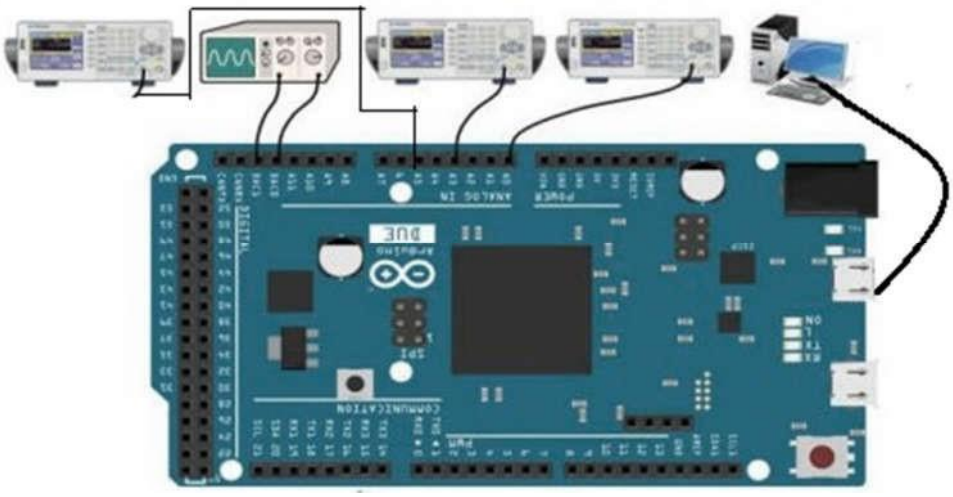

Figure 21 : Hardware configuration model of the separation system. C.

C. Discussion.

The results of the implementation show an operation similar to that obtained by simulation on Simulink, which confirms the precision of the SOBI method that we used to linearly separate 03 signals by a matrix A. our model allows the separation of three signals, however we selected three observable signals via the hypotheses of our internal similarity function block among the 9 expected signals, because the latter being morphologically closer to those expected. The separation performance was evaluated using the signal to signal ratio. interference (SIR), and the performance index (PI) recorded in the table below.

Table 3: Performance tables of the algorithm used

Algotithm.	**Execution Mode.**	**PI**	**SIR**
SOBI	Simulation on Matlab	0,0012	78,56
	Real-time implementation on Arduino Due.	0,0019	80,12

D. Implementation results.

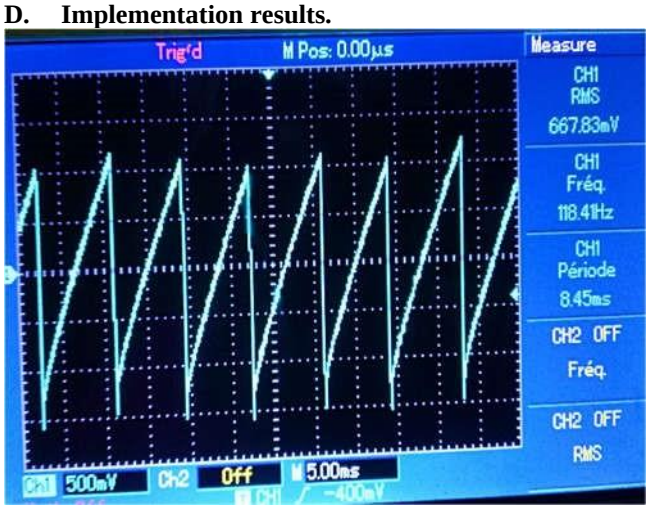

Figure 22 : Source signal S1

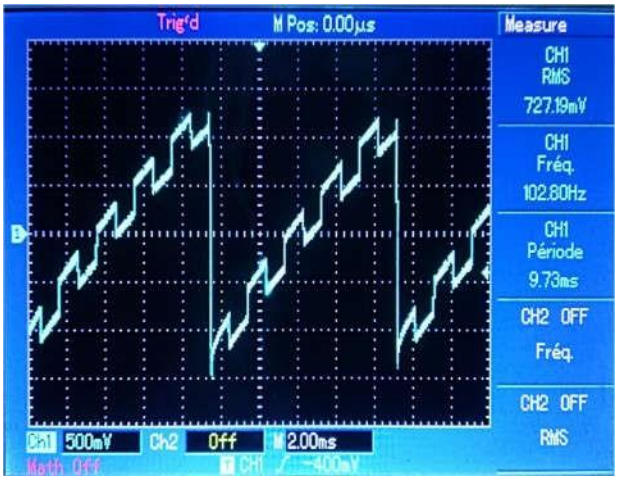

Figure 23 : Mixed signal X1

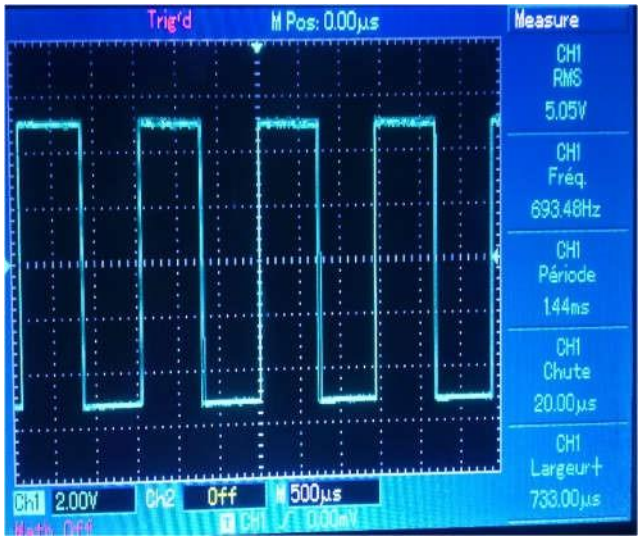

Figure 24 : Signal source S2

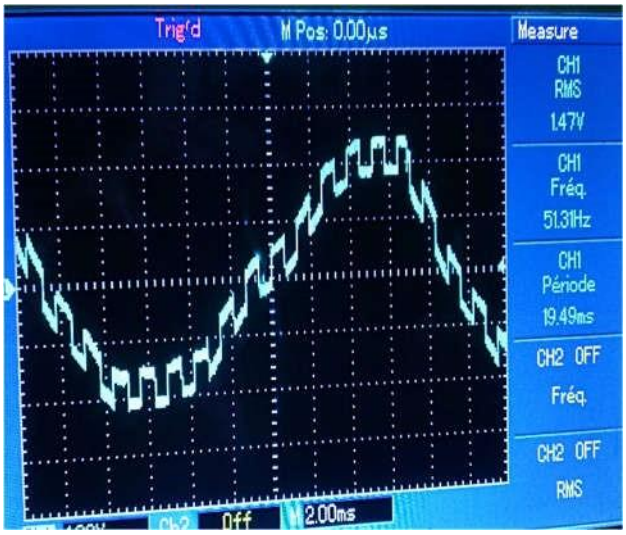

Figure 25 : Mixed signal X2

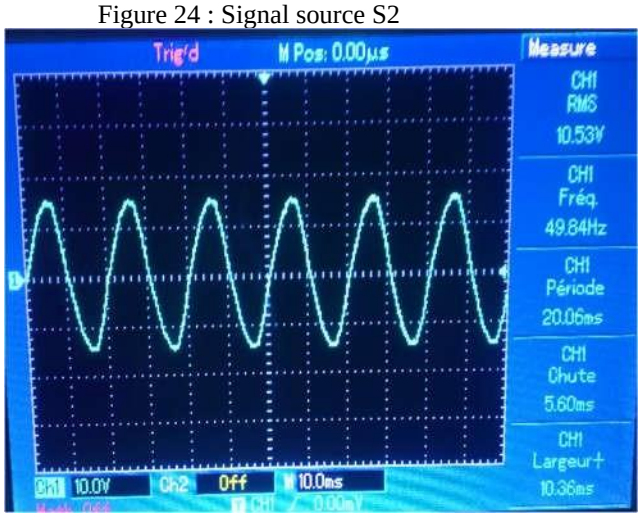

Figure 26 : Source signal S3.

Figure 27 : Mixed signal X3

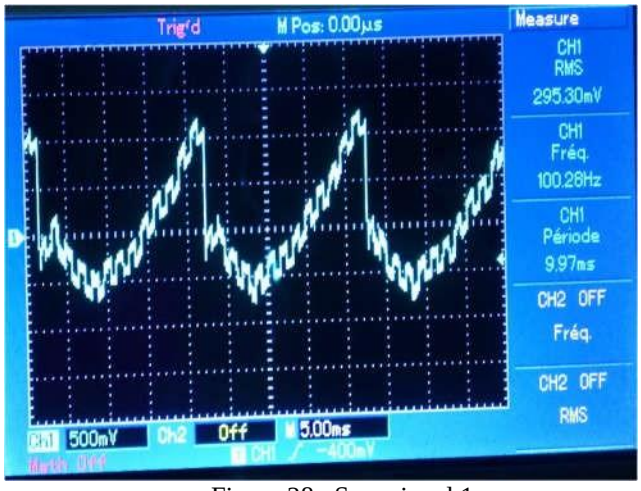

Figure 28 : Sum signal 1

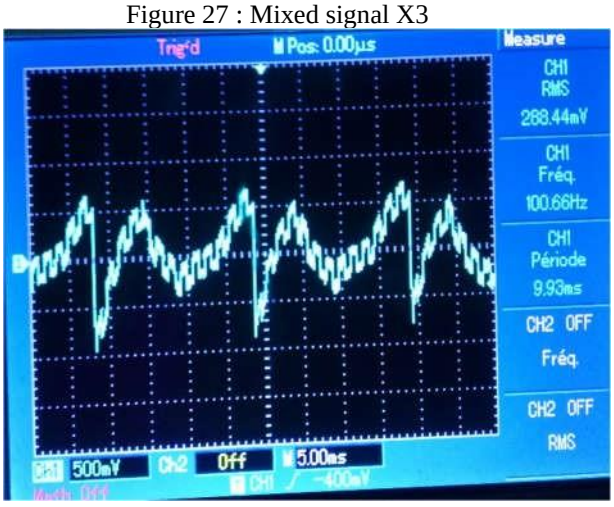

Figure 29 : Sum signal 2.

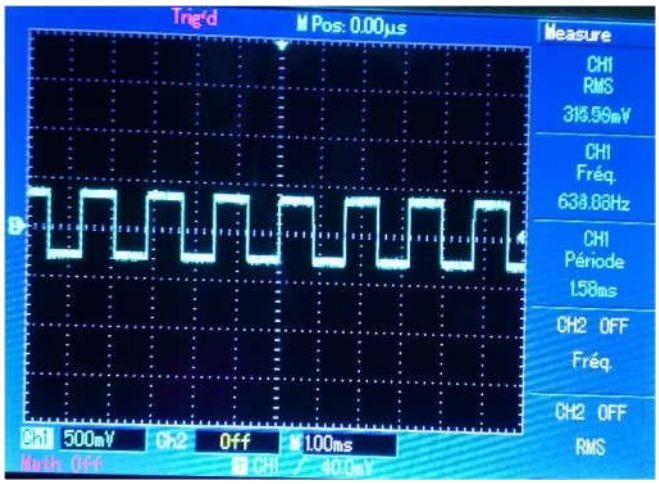

Figure 30Estimated signal Y1.

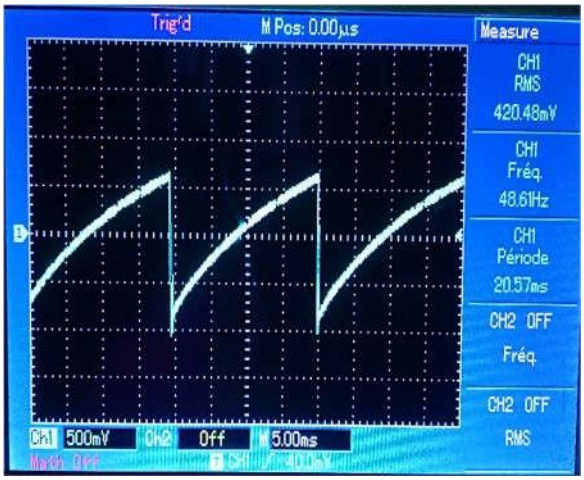

Figure 31 : Estimated signal Y2.

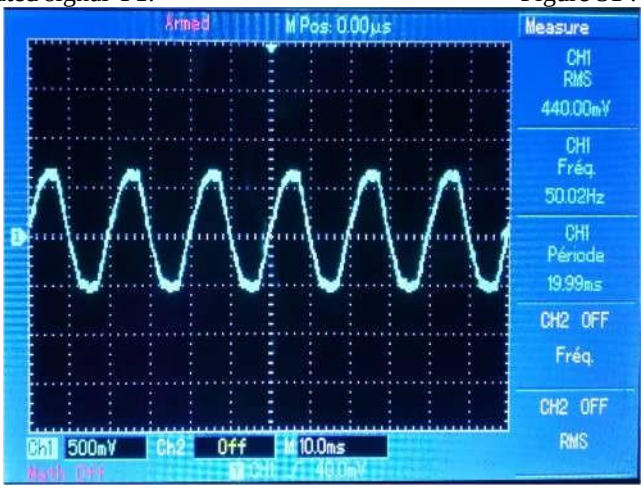

Figure 32: Estimated signal Y3

4. CONCLUSION

In this work, we presented the possibility of blind separation of three sound signals in the Matlab/Simulink software. To validate the performance and operation of our SOBI algorithm in practice and in real time, we implemented this algorithm in the Arduino Due board as free and low-cost materials, using three GBF generators and an oscilloscope. We obtained precise results similar in form to those of the simulation in Matlab. The performance indices and signal/interference ratios confirmed our conclusion with a satisfactory result.

REFERENCES

[1]. The Society of Obstetricians and Gynaecologists of Canada, Surveillance du bien-être fœtal : Directive consensus d'antepartum et intrapartum, 2007.

[2]. H. Rey, E. T. Bowe, L. S. James, Impact of Fetal Heart Rate Monitoring & Blood Sampling on Infant Mortality & Morbidity - Ongoing Study, Pediatric Research 8 (4) (1974) 450. 17

[3]. Programme National Multisectoriel de Lutte contre la Mortalité Maternelle, Néonatale et Infanto-Juvénile 2014- 2020 au Cameroun.

[4]. F.-M. Chang, K.-F. Hsu, H.-C. Ko, B.-L. Yao, C.-H. Chang, C.-H. Yu, R.-I. Liang, H.-Y. Chen, Fetal Heart Volume Assessment by Three-Dimensional Ultrasound, Ultrasound Obstetrics and Gynecology 9 (1) (1997) 42–48.

[5]. J. Jezewski, J. Wrobel, K. Horoba, Comparison of Doppler Ultrasound and Direct Electrocardiography Acquisition Techniques for Quantifica tion of Fetal Heart Rate Variability, IEEE trans. on Biomedical Engi neering 53 (5) (2006) 855–864.

[6]. A. de Cheveigne, H. Kawahara, YIN, a Fundamental Frequency Estimator for Speech and Music, J. of the Acoustical Society of America 111 (4) (2002) 1917–1930.

[7]. C. Jutten et J. Herault, «Space or time adaptive signal processing by neural network models», AIP Conference Proceedings, p. 206-211, 1986.

[8]. C. Jutten et A. Taleb, Source Separation: From Dusk Till Dawn, 2000.

[9]. C. Jutten et J. Herault, «Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture», Signal Processing, vol. 24, n° 11, pp. 1-10, 1991.
[10]. P. Comon, «Independent component analysis, A new concept? », Signal Processing, vol. 36, n° %13, pp. 287-314, 1994.
[11]. M. Zhang, M. Zhu et W. Ma, « Implementation of FastICA on DSP for Blind Source Separation », Procedia Engineering, vol. 29, pp. 4228-4233, 2012.
[12]. U. Meyer-Baese, C. Odom, G. Botella et A. Meyer-Baese, «Independent component analysis algorithm FPGA design to perform real-time blind source separation», Independent Component Analyses, Compressive Sampling, Large Data Analyses (LDA), Neural Networks, Biosystems, et Nanoengineering XIII, vol. 9496, 2015.
[13]. V. Singh, V. Somani et J. Manikandan, « FPGA implementation of blind source separation using a novel ICA algorithm », IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), pp. 67-71, 2017.
[14]. J. Pardey, M. Moulden, C. W. G. Redman, A Computer System for the Numerical Analysis of Nonstress Tests, Am. J. of Obstetrics and Gynecology 186 (5) (2002) 1095–1103.
[15]. Mekhfioui, R. Elgouri, A. Satif, L. Hlou . Arduino Due Implementation of an Algorithm for Blind Source Separation using Matlab Simulink M. International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075 (Online), Volume-9 Issue-2, December 2019
[16]. H. Arahmane, E. Hamzaoui and R. Cherkaoui El Moursli, «Neutron Flux Monitoring Based on Blind Source Separation Algorithms in Moroccan TRIGA MARK II Reactor,» Science and Technology of Nuclear Installations, vol. 2017, 2017.
[17]. C. Wang, J. Wang et T. Zhang, «Operational modal analysis for slow linear time-varying structures based on moving window second order blind identification,» Signal Processing, Vols. %1 sur %2169-186, p. 133, 2017.
[18]. «Arduino Support from MATLAB,» MathWorks, Inc, [En ligne]. Available: https://www.mathworks.com/hardwaresupport/arduino-matlab.html. [Accessed 04 01 2023].
[19]. M. Mekhfioui, R. Elgouri,A. Satif, L. Hlou , <<Arduino Due Implementation of an Algorithm for Blind Source Separation using Matlab Simulink>> (IJITEE) ISSN: 2278-3075 (Online), Volume-9 Issue-2, December 2019.
[20]. Rumana Islam and Mohammed Tarique <<Blind Source Separation Of Fetal ECG Using Fast Independent Component Analysis And Principle Component Analysis >>, ISSUE 11, NOVEMBER 2020 ISSN 2277-8616
[21]. Mohcin Mekhfioui, Rachid Elgouri, Amal Satif, Laamari Hlou <<Real-Time Implementation Of A New Efficient Algorithm For Source Separation Using Matlab & Arduino Due >>ISSUE 04, APRIL 2020 ISSN 2277-8616.
[22]. N. Shirazi, A. Walters, and P. Athanas, "Quantitative analysis offloating point arithmetic on FPGA based custom computing machines," in Proc. IEEE Symp. FPGAs Custom Comput. Mach., 1995,pp. 155–162.
[23]. S. Ding, «A Power Iteration Algorithm for ICA Based on Diagonalizations of Non-Linearized Covariance Matrix,» First International Conference on Innovative Computing, Information and Control, Beijing, 2006, pp. 730-733.
[24]. Ying Tan, Member, IEEE, Jun Wang, Senior Member, IEEE, and Jacek M. Zurada, Fellow, IEEE ; << Nonlinear Blind Source Separation Using a Radial Basis Function Network>> IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 12, NO. 1, JANUARY 2001.
[25]. Mohcin Mekhfioui, Rachid Elgouri, Amal Satif, Laamari Hlou. <<A Comparative Approach of Blind Source Separation with Arduino Due and TMS320C6713>> Research, 8(3), March 2020, Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter42832020.pdf