ISSN 2347 - 3983
Volume 8. No. 3, March 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter42832020.pdf
https://doi.org/10.30534/ijeter/2020/42832020

A Comparative Approach of Blind Source Separation with Arduino Due and TMS320C6713

Mohcin Mekhfioui[1], Rachid Elgouri[2], Amal Satif[2], Laamari Hlou[1]
[1]Laboratory of Electrical Engineering and Energy Systems, Faculty of Science, Ibn Tofail University, Morocco, mohcin.mekhfioui@uit.ac.ma
[2]Laboratory of Electrical Engineering and Telecommunications Systems, ENSA, Ibn TofailUniversity,Morocco

ABSTRACT

This paper presents a comparison between two blind source separation methods: the first one based on a microcontroller board such as Arduino Due by the RTW (Real-Time Workshop) integrated into Matlab Simulink and the second one based on a DSP board (TMS320c6713). This study compared the space occupied, the execution speed and the separation index between the two platforms Arduino and DSP. The separation was practically simulated on the Arduino Due and TMS320c6713 board by two signal generators and an oscilloscope.

Key words : Blind Source Separation, Arduino Due, TMS320C6713, SOBI.

1. INTRODUCTION

The Blind Source Separation (BSS) is a rapidly developing discipline in the signal processing community that offers a solution for separating mixed signals during their propagation. The BSS, therefore, consists in estimating a set of unknown source signals from a set of observed signals that are mixtures of these source signals. Without any additional hypothesis, the problem carried out by the BSS is a wrong problem posed. This is why all BSS methods have assumptions about both sources and mixing.

Source separation techniques have been applied in various scientific and technological fields, such as telecommunications, acoustics, imaging and biomedical signal processing.

The importance of this separation method justifies the different use of platforms such as Arduino Due[1], DSP [2] and FPGA [3]-[4].The Arduino Due is an open source electronic prototyping platform based low cost on the ARM Cortex-M3 microcontroller [5].

This work presents a comparative study between the implementation of the SOBI (Second Order Blind Identification) [2] algorithm for the separation of two source signals in the Arduino Due board and in the TMS320c6713 board, using Matlab/Simulink and Code Composer Studio (CCS)[6].

This work is organized as follows: Section II presents the principle of blind source separation and its mathematical model, the Arduino Due board and the TMS320C6713 board, Section III presents studies the execution speed and the separation index between the two platforms. Section IV shows some results and analysis. Finally, concluding remarks are given in Section V.

2. MATERIAL AND METHODOLOGY

2.1 Blind Source Separation

A The main objective of the blind source separation (BSS) is to estimate a set of unknown source signals s(t) using only the mixed signals obtained by a series of sensors x(t), also called "observations" Fig. 1 [7]-[8].

The term Blind denotes the fact that there is no a priori information about the sources or how they have been mixed. In the general case, three conditions are necessary to perform this technique[9]:

- The sources are statistically independent;
- The number of sensors is higher or equal to the number of sources;
- A mixing matrix between the sources and the sensors.

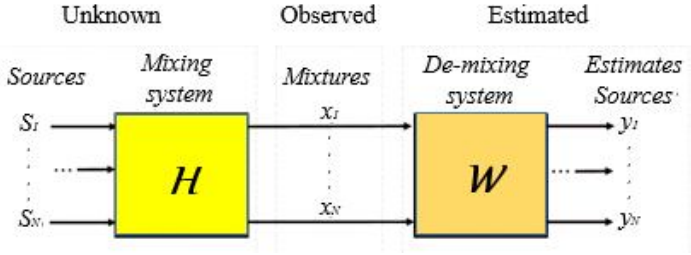

Figure 1: Principle of the blind sources separation

2.2 Mathematical Model

The basic model of the blind source separation (BSS) depends on the assumed mixing model, in the general case the

equation of the instantaneous linear mixing is written as follows[9]:

$$x_i(t) = \sum_{j=1}^{n} a_{ij} s_j(t) + v_i(t) \quad (1)$$
$$i = 1, 2, \dots, m \text{ and } j = 1, 2, \dots, n$$

Where:
$x(t) = [x_1(t), \dots, x_m(t)]^T$ is the vector containing the observations;
a_{ij} is the mixing system;
$s(t) = [s_1(t), \dots, s_n(t)]^T$ is the vector containing N signals emitted by N unknown sources;
v(t) is an additive noise vector we consider it negligible in the rest of the calculations.
The matrix form of equation (1) is written as follows:

$$X(t) = HS(t) \quad (2)$$

Where H is a mixture matrix of size $n \times m$
It is about finding in the ideal case the matrix W of size $n \times m$ which inverts the mixture and provides the output vector:

$$y(k) = W\,x(k) = W\,H\,s(k) \approx s(k) \quad (3)$$

The estimated sources are given by the vector $s(k)$ and their corresponding projections for the different microphones are given by the estimated matrix: $H = W^{-1}$.

2.3 The DSK TMS320C6713 board

The TMS320C6000 platform of the DSP digital signal processors is a part of the TMS320 own family of TEXAS devices. It includes the TMS320C62x fixed arithmetic and TMS320C67x floating arithmetic processors. The TMS320C6713 is taken into consideration the most efficient member of the C67x class, its VLIW architecture lets in it to process 8 instructions in parallel by means of 8 functional units. The discussion in this phase will consciousness at the TMS320C6713 processor. The structure and related devices may also be mentioned[10], Fig 2.

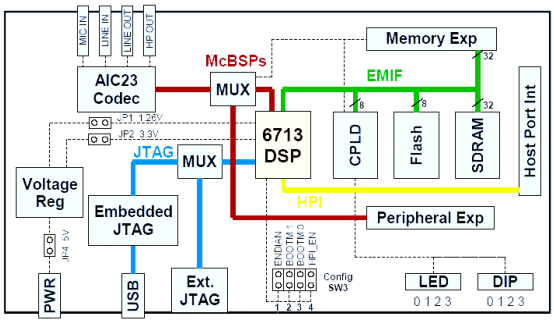

Figure 2:TMS320C6713DSK Physical layout

The TMS320C67x DSP generation is supported by the eXpress DSP IT reference development toolkit, including a highly optimized C/C++ compiler, Code Composer Studio integrated development environment (CCSIDE), JTAG-based emulation and real-time debugging, and the DSP/BIOS kernel. CCS offers solid central functions with user-friendly configuration and graphical visualization tools for system design. C/C++ programming for the application is respected, linked and executed by the CCS.

To use CCS with Matlab, you must install the Simulink Support Package Library for Texas instruments C6000. all implementation steps start by generating the Simulink model for the system using the Simulink blocks in MATLAB until the file download '*.*out*'on the DSP card[2], shown in Fig. 3.

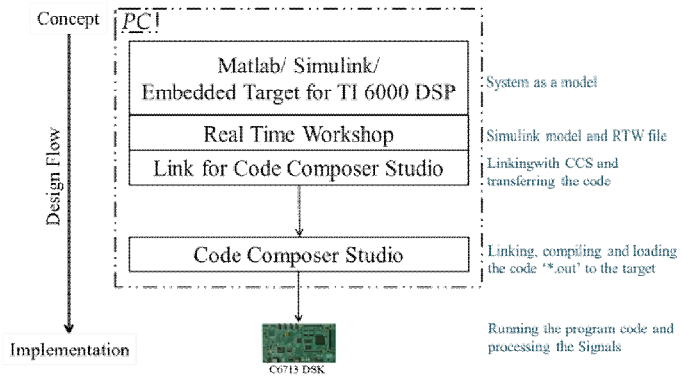

Figure 3: Flow diagram of the procedures of implementing the model on C6713 DSK

2.4 C. Arduino Due card

The Arduino Due is a programmable card based on a low-cost ARM Cortex-M3 microcontroller[11]. It contains 54 digital pins (including 12 PWM), an 84 MHz clock, 4 UARTs, 12 analog inputs, a JTAG header Fig.4.[12]

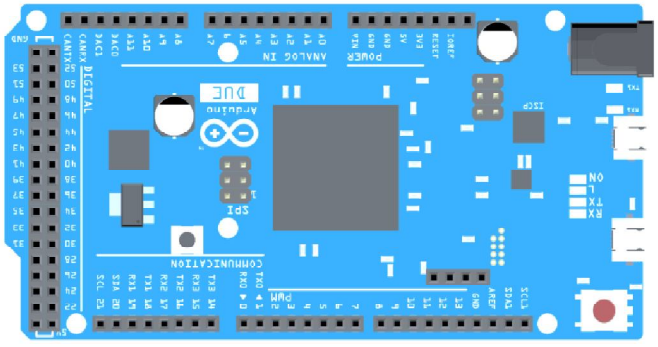

Figure 4: Arduino Due card

To use the Arduino block in Matlab, you must install the Simulink Support Package Library for Arduino hardware that can be obtained and installed by clicking on the Add-On on Matlab. When the installation is complete, the library can be added to the Simulink Library Browser as a Simulink support package for Arduino hardware.

Matlab Simulink [13]wishes to be configured to be operated with Arduino Due. Within the simulation alternatives, the external mode is enabled to run in real-time with the external

hardware. While execution is requested, RTW converts the Simulink block diagram into C/C++ code that allows you to assemble and download the code into the microcontroller. The code is run in actual time with the chosen sampling time within the blocks and the selected signals are sent to Simulink via USB interface. Figure 5 displays the diagram of the real Time Workshop[14].

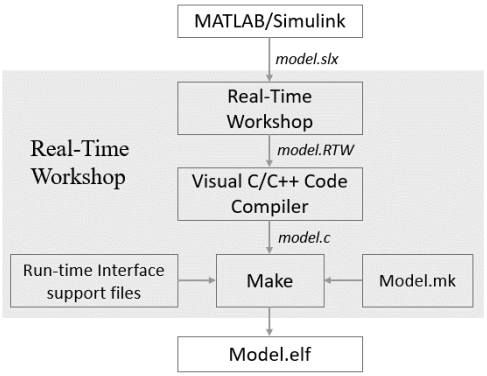

Figure 5: Flow diagram of the procedures of implementation the model on Arduino Due

3. PROPOSED WORK& RESULTS

The objective of this study is to present a comparison between the implementation of the SOBI algorithm for blind source separation on the Arduino Due board and the TMS320C6713 board. The separation algorithm will be programmed in R language (Matlab) and Simulink blocks, such as Texas instrument C6000 blocks for the TMS320C6713 board and Arduino package blocks for the Arduino Due board.
The flowchart of our comparison is presented in fig. 6, it is based on two Arduino and DSP programming cards for the separation of two multiplexed sources.

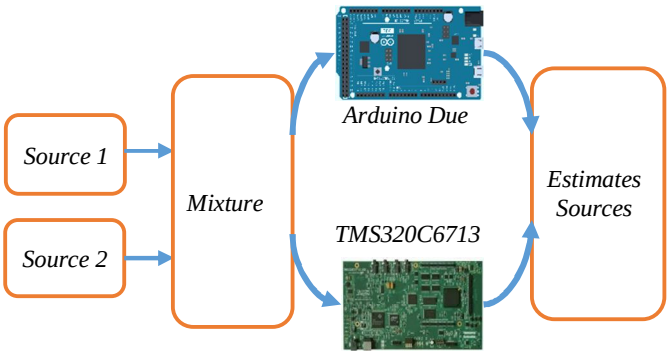

Figure 6: A typical use of Arduino due and TMS320C6713 with the Blind source separation algorithm.

Our comparison method is based on the calculation of execution speed, the memory space used, the price of the material and the precision of separation. For the validation of the results in real-time we used two signal generators GPF in order to create two source signals (s1 and s2) of the same shape and different frequency, these two signals will be mixed by a matrix A to produce the two mix signals x1 and x2.
The following table shows the comparison results between the Arduino Due card and the TMS320c6713 card for some parameters, such as execution speed, memory space used and performance index, Table 1.

Table 1: Comparison between Arduino Due and TMS320c6713 for some parameters

	TMS320C6713	ARDUINO DUE
Memory space used	280 KB	322 KB
Clock Speed	225 MHz	84 MHz
Analog conversion	32 bits	12 bits
Performance Index (PI)	0,0012	0,0018
Price	500 USD	30 USD

The different input and output signals of the SOBI algorithm implemented in the Arduino Due and TMS320C6713 board are presented in Fig. 7 and Fig. 8.

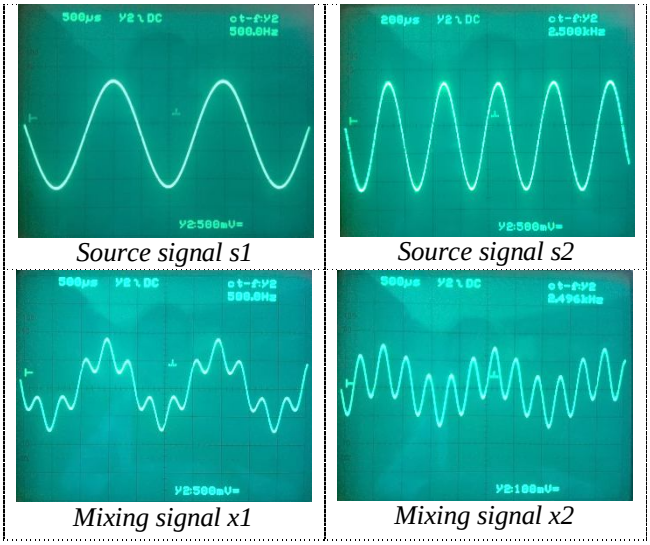

Figure 7: Sources signal and mixing signal used in our study

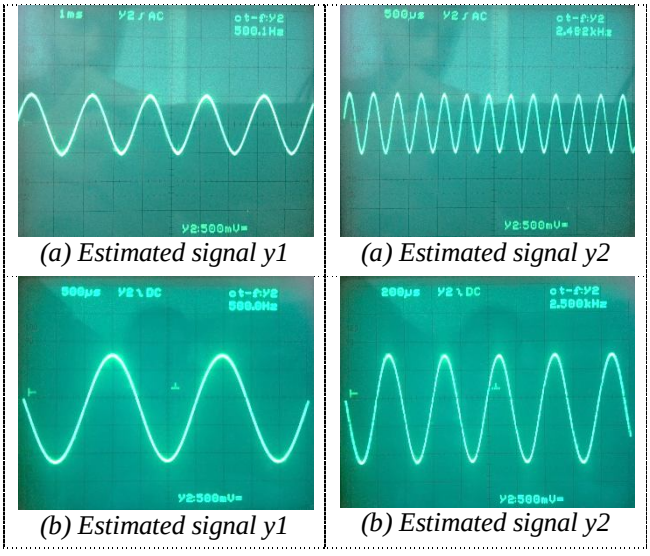

Figure 8: Result of implementation using the SOBI Algorithm with Arduino due (a) and TMS320C6713 (b)

4. DISCUSSION

The experimental simulation results show that blind source separation with the SOBI algorithm has high performance in terms of separation for the two boards. The performance of the TMS320C6713 board presented by various simulations and applications. Some of the most important results are the conversion precision performed by the AIC23 codec of the board with 32bits, the execution speed with its 255 MHz clock. On the other hand, the Arduino Due board also shows a great performance for basic applications in terms of separation and memory space used, without forgetting its lower cost and its open-source hardware.

5. CONCLUSION

The implementation of the blind source separation algorithm using Arduino Due is designed for low-frequency projects that do not require large memory.
However, this separation based on the TMS320C6713 board offers a large range of space to create processing blocks and completely embedded recordings. As well as high-speed data processing for audio applications.
The design has been implemented on Arduino Due and DSK TMS320C6713 devices, their comparison summaries are displayed.

REFERENCES

[1] M. Mekhfioui, R. Elgouri, A. Satif, et L. Hlou, « **Arduino Due Implementation of an Algorithm for Blind Source Separation using Matlab Simulink** », *Int. J. Innov. Technol. Explor. Eng. IJITEE*, vol. 9, n° 2, déc. 2019, doi: 10.35940/ijitee.B7385.129219.

[2] M. Mekhfioui *et al.*, « **A Comparative Study and Implementation of Blind Source Separation Algorithm using MATLAB and TMS320c6713 DSK** », *J. Eng. Appl. Sci.*, vol. 15, n° 5, p. 1074-1081, déc. 2019, doi: 10.36478/jeasci.2020.1074.1081.

[3] V. Singh, V. K. Somani, et J. Manikandan, « **FPGA implementation of blind source separation using a novel ICA algorithm** », in *2017 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, 2017, p. 67□71, doi: 10.1109/ICCE-ASIA.2017.8307845.

[4] M. Mekhfioui, R. Elgouri, A. Satif, et L. Hlou, « **Real time hardware co-simulation for blind image separation algorithm using ZYNG 7000 & xilinx system generator** », *Int. J. Emerg. Trends Eng. Res.*, vol. 8, n° 2, p. 365-371, 2020.

[5] M. Ghosh, S. Basu, S. Pandit, et S. Barman (Mandal), « **Design of a Health Monitoring System for Heart Rate and Body Temperature Sensing Including Embedded Processing Using ARM Cortex M3** », in *Computational Intelligence in Pattern Recognition*, Singapore, 2020, p. 93-103, doi: 10.1007/978-981-13-9042-5_9.

[6] S. Qureshi, ***Embedded Image Processing on the TMS320C6000TM DSP: Examples in Code Composer StudioTM and MATLAB***. Springer Science & Business Media, 2005.

[7] J. E. Song, J. Shin, H. Lee, H. J. Lee, W.-J. Moon, et D.-H. Kim, « **Blind Source Separation for Myelin Water Fraction Mapping using Multi-echo Gradient Echo Imaging** », *IEEE Trans. Med. Imaging*, p. 1□1, 2020, doi: 10.1109/TMI.2020.2967068.

[8] O. Cherrak, H. Ghennioui, N. Thirion-Moreau, et E. H. Abarkan, « **Blind separation of complex-valued satellite-AIS data for marine surveillance: a spatial quadratic time-frequency domain approach** », *Int. J. Electr. Comput. Eng. 2088-8708*, vol. 9, n° 3, p. 1732-1741, juin 2019, doi: 10.11591/ijece.v9i3.pp1732-1741.

[9] A. Cichocki et S. Amari, « **Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications** », 2002.

[10] Texas Instruments, « **TMS320C67x/C67x+ DSPCPU and Instruction Set Reference Guide** ». pp. 19-25,2006.

[11] S. Bestley Joe, R. Ramadevi, V. Amala Rani, et G. Rajalakshmi, « **Automatic Cooking Machineusing Arduino** », *Int. J. Emerg. Trends Eng. Res.*, vol. 8, n° 1, p. 35-40, 2020.

[12] « **Arduino Due | Arduino Official Store** ».[Online]. Available: https://store.arduino.cc/arduino-due. [Accessed: 10-03-2020].

[13] M. A. Aaron Don, B. Lourdes Racielle, Z. M. Matthew, et N. Isabel, « **Binary Phase Shift Keying Simulation with MATLAB and SIMULINK** », *Int. J. Emerg. Trends Eng. Res.*, vol. 8, n° 2, p. 288-294, 2020.

[14] « **Arduino Support from MATLAB** ». [Online]. Available : https://fr.mathworks.com/hardware-support/arduino-matlab.html. [Accessed: 10-02-2020].