IMPLEMENTATION OF BLIND SOURCE SEPERATION USING FPGA

Anu vyshakh C.P, Vineetha jain K.V, Dr.Ramesh Chinthala(Dept of ECE)

Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru,
Amrita Vishwa Vidyapeetham, India
anuvyshakhcp@gmail.com, _jain_vineetha@blr.amrita.edu, c_ramesh@blr.amrita.edu

Abstract — " Blind source separation is a method to separate source signals from mixed signals without any data loss". This technique has recently drawn a lot of attention in the field of Space research. This paper presents an efficient implementation of the Blind source separation algorithm [FASTica] using FPGA and its optimization. In general independent component analysis[ICA] algorithm separates the source signal from mixed-signal by finding a linear transformation that can maximize the mutual independence of the mixture. FASTica algorithm is the advanced version of the ICA algorithm.FASTica algorithm measures the non-gaussianity using the kurtosis algorithm to find the independent source of their mixture. Blind source separation algorithms are highly computational complex in nature[Vector multiplication, matrix multiplication, Matrix inverse, etc.], To provide high computing power for the algorithm while implementing we are using hardware called FPGA, which is hard to achieve in MCUs [Microcontroller units]. "FPGAs are semiconductor devices which contain programmable logic blocks and interconnection circuit". The main challenges while hardware implementation of BSS algorithms is required high processing speed [Because of complex computations], Memory requirement, and power consumption. These challenges can easily solve by using FPGA. Optimization of a blind source separation algorithm can be done by using the technique called pipelining, Through this, the performance and accuracy of blind source signal separation are improved.

Keywords—FPGA, Redpitaya, ADC, DAC, Blind source separation, FASTica, Memory.

I. INTRODUCTION

Blind source separation[BSS] is a technique to unmix the source signal from mixed signals without any data loss. BSS is intensively using in different fields such as acoustics, biomedical, signal processing, geophysics, image processing, statistics, etc.

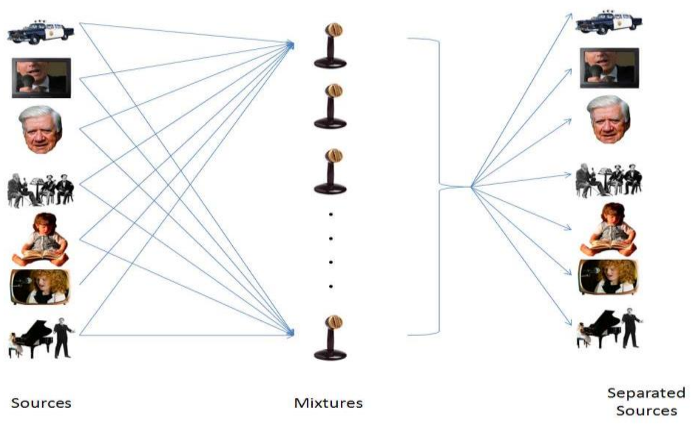
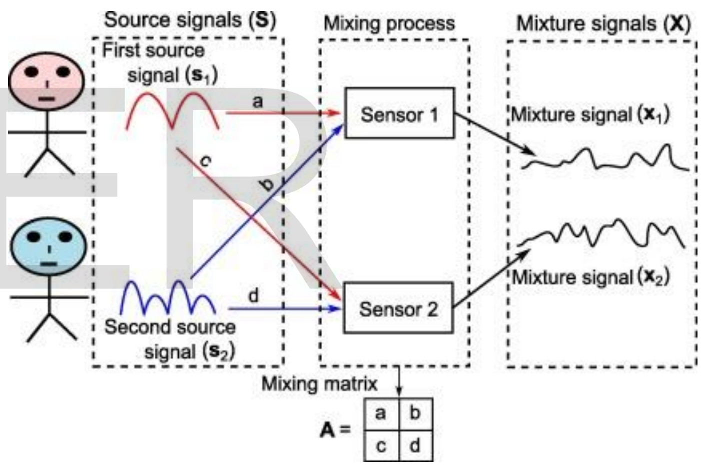


Fig.1 Blind source separation.

Different types of techniques are using to separate the source signal from its mixed signals [Blind source separation] such as "principal components analysis, Singular value decomposition, Independent component analysis, Dependent component analysis, Non-negative matrix factorization, Low-complexity coding and decoding, Stationary subspace analysis, Common spatial pattern",etc. Among these independent component analyses [ICA] algorithm drawn a lot of attention in radar communication, space research, and the medical field. In general ICA algorithm separates the source signal from mixed signals by finding a linear transformation that can maximize the mutual independence of the mixture. A classical example of an independent component analysis is a cocktail party problem.



\

Fig.2 ICA[Independent component analysis]

In a cocktail party problem, different audio signals are generating from different audio sources. Then receiving the audio signal through different sensors/Transducers like microphones. The captured audio signals by the microphones will be mixed audio signals received from different sources. Then the next step is to separate the mixed audio signal received by the different microphones. For the separation of mixed audio signals, an advanced blind source algorithm is using called ICA[independed component analysis]. FASCIA is the advanced version of independent component analysis which measures the non-guassianity using kurtosis algorithm to find an independent source of their mixtures. Different challenges are there to implement in hardware such as the speed of the system, bandwidth requirements, segregation of raw data (not classified), how to process and transfer data fast, etc. Blind source separation algorithms are highly computational complex in nature [Vector multiplication, matrix multiplication, Matrix inverse, etc.]. To provide high computing power for the algorithm we are using FPGA, Which is hard to achieve in MCUs[Microcontroller units].

"FPGAs are semiconductor devices that contain programmable logic blocks and interconnection circuits". In general, FPGAs are more attractive in the area of signal processing because of its high processing speed and less power consumption. Additional advantages of FPGAs are its reprogramming ability, memory [according to our requirement], parallelism [more algorithms can run at a time], etc. The main challenges while hardware implementation of BSS algorithms is required high processing speed [Because of complex computations], Memory requirement, and power consumption. These challenges can easily solve by using FPGA Field programmable gate array (FPGA).

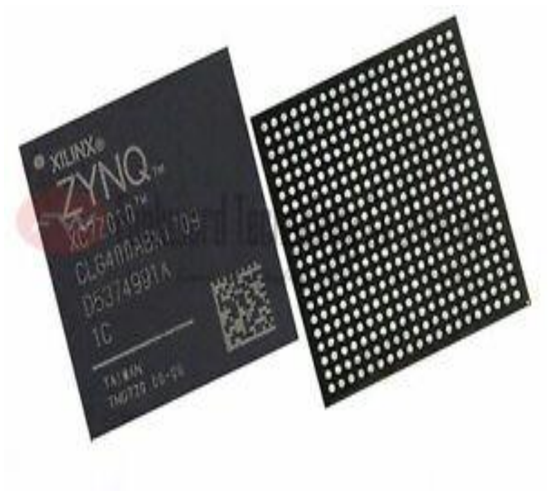


Fig.3 Xilinx ZYNQ 7010 SOC [FPGA]

"The Zynq®-7000 family is based on the Xilinx SoC architecture. These products integrate a feature-rich dual-core or single-core ARM® Cortex™-A9 based processing system (PS) and 28 nm Xilinx programmable logic (PL) in a single device". The ARM Cortex-A9 CPUs are the heart of the PS and also include on-chip memory, external memory interfaces, and a rich set of peripheral connectivity interfaces
Blind source separation is a method to separate mixed signals to its source signals. Major drawbacks of Blind source algorithms are its output correlation of noise and the inaccuracy of output separated source signals. Our proposed method is to separate the mixed-signal which is coming from unknown sources to its initial source signals with the help of an optimized algorithm[FASTICA] and its hardware implementation in FPGA to improve the speed of the system. Regarding this application, separate blocks of an FPGA can be reprogrammed to fit the requirements of blind source signal separation algorithm. Through this, the performance and accuracy of blind source signal separation can be improved.

II. LITERATURE SURVEY

[1]For adaptive noise cancellation and blind source separation[BSS] they are using an algorithm called Independent component analysis[ICA] and the algorithm is dumping into an FPGA.ICA algorithm requires high computational power, to provide high computing power for this algorithm, a special digital processor is designed and implemented in FPGA. A mixed input analog signal is given to the PUT terminal of FPGA and converted this to a digital signal with the help of ADC for processing in FPGA. In future work Fpga implementation of ICA algorithm is integrating with existing speech recognition chip for getting a precise and efficient system [2] This one is dealing with blind source separation of fetal ECG using FPGA. The algorithm that is used for blind source separation [BSS] is the TORKKOLAS BSS algorithm. This an advanced version of the ICA algorithm. Here They are using FPGA architecture to handle the high computational load of BSS(Blind source algorithm)The FPGA using is SPARTAN-3. For DSP algorithms an idle target rapid prototype is offering by FPGA. They have designed a MAC (multiply-accumulate) unit to reduce the complexity of logic blocks. It efficiently reduces the number of multipliers and adders (reducing logic block complexity). Through this, the power consumption can be reduced and the performance of the system is increasing.

[3]This paper is about the separation of heartbeat sound and its FPGA implementation. The technique which is using to capture heartbeat is Auscultation. Fascia is using as a blind source separation algorithm and dumping that algorithm in FPGA. With the help of 4 channel analog preamplifier and 4 channel sound card(outside FPGA), they are amplifying low amplitude heart signal a giving the input to the put terminal of FPGA. The FPGA board they are using VIRTEX-2 PRO. Advantages of the board are Block ram components, Fast detected multipliers, Fast select ram, SRLs. The output(separated signal) is taking out with the help of DAC and for experimental purpose showing it in CRO.[4] In this paper, they are using neural networks for face detection and implemented in FPGA. To represent the number system they have used a floating-point algorithm. This helps reduction in memory, so it is a better option for the neural network. The performance of FPGA based system is 38 times faster than pc(Pentium 4).this speed helps to have a better picture than preprocessed. Multiplier and adder are 2 modules in FPU units, Using FPGA they used a hard IP core for a multiplier of FPU multiplication module to improve speed. They Have used XILINX XCS51500, it gives a more reasonable price area than vertex series.

[5] Independent component analysis[ICA] is a well-known technique for blind source separation. The advanced version of the ICA algorithm is a FAST independent component analysis [FASTica]. implementing the FASTica algorithm in an FPGA[virtex-5] at a sampling rate of input signal 192 kHz. The floating-point algorithm is providing goof accuracy and performance comparing to the fixed point. Due to the high computational power of FASTica [without optimization] algorithm and its implementation in FPGA taking more consuming. A major drawback of this paper is not included in any method to simplify the FASTica algorithm and hard optimization techniques also.[6] Model of Fiber Optic Transmission System Based on the Red Pitaya analyzing the possibility of chromatic scattering and attenuation simulation. The final design of the model has been implemented into a rediptaya board. The major advantages of redpitaya boards are its high bandwidth for receiving input signals, dual-core Cortex A9 processor, Zinq 7010 Soc FPGA, memory availability, and automated in nature. The conclusion of this paper according to our work is the performance of redpitaya board and signal processing ability.[7] "A Single-Chip FPGA Design for Real-Time ICA-Based Blind Source Separation Algorithm" is specially implemented for separating the mixer of the audio signal in the real-world different sensor [Multiple] applications. Here dumping the blind source algorithm in a low-cost FPGA. The hardware optimization technique called MAC [multipin and accumulate unit] improves hardware efficiency. But instead of the floating-point algorithm, they are using a fixed-point algorithm which reduces the output accuracy is one of the drawbacks.

III. PROPOSED WORK

Blind source separation is a method to separate mixed signals to its source signals. Major drawbacks of Blind source algorithms are its output correlation of noise and the inaccuracy of output separated source signals. Our proposed method is to separate the mixed-signal which is coming from unknown sources to its initial source signals with the help of an optimized algorithm[FASTICA] and its hardware implementation in FPGA to improve the speed of the system. Regarding this application, separate blocks of an FPGA can be reprogrammed to fit the requirements of blind source signal separation algorithm. Through this, the performance and accuracy of blind source signal separation can be improved.

Optimization of FASTICA algorithm

In general independent component analysis[ICA] algorithm separates the source signal from mixed-signal by finding a linear transformation that can maximize the mutual independence of the mixture. For this need to develop a separating matrix W. This method is difficult to sort out the gradient information of the independent sources. So instead of conventional gradient descent method, introducing a new genetical algorithm for the optimization of algorithm.

SYSTEM SPECIFICATION:

The hardware required for this work is a Redpitaya bord which is having a Zinq 7000 FPGA. Redpitaya is an "All programmable SoC with an ARM-based processor supporting hardware programmability of an FPGA. Suited for unique application requirements".Redpitaya stem bord consist of dual-core arm cortex A9 processor, Zinq 7010 system on chip FPGA, two input and output ports, RAM, Extension connectors, Jumper input voltage, Syncconnectors, Micro USB(power), Micro SD slot, micro USB (console), USB, Ethernet channel, etc.

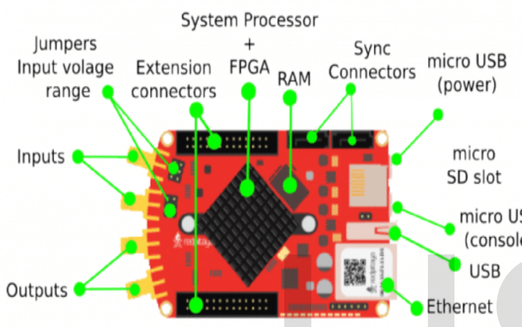
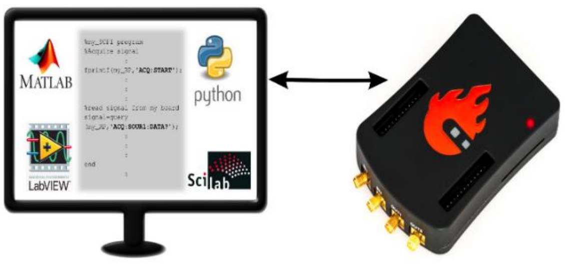


Fig.4 Redpitaya stem lab.

Redpitaya stemlab consists of the Dual-core arm cortex A9 processor. It is a 32-bit multicore processor with power and performance-optimized. The major attraction of the cortex A9 processor is its dual-issue, partially out-of-order pipeline. Armv7-A is the architecture of cortex A9. **"**The Zynq®-7000 FPGA family is based on the Xilinx SoC architecture".This product contains both ARM® Cortex™-A9 based processing system and Xilinx programmable logic of 28nm in one device. The Zynq®-7000 consists of I/O Peripherals and Interfaces, Caches, ARM Cortex-A9 Based Application Processor Unit (APU), External Memory Interfaces, Interconnect, On-Chip Memory, etc.

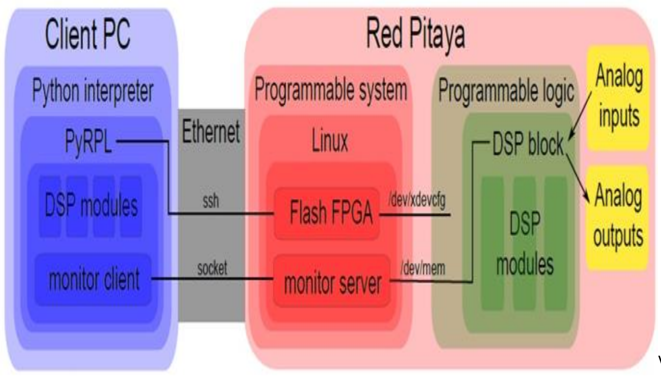
Red pitaya stem lab drew a lot of attention in the field of signal processing. It consists of zinq 7010 FPGA and its advantage is its prewritten modules. Different platforms are there to write programs like Jupiter book, pyrpl, Matlab, Scilab, Python, etc
.



```
%my_SCPI program
%Acquire signal
        :
fprintf(my_RP,'ACQ:START');
        :
        :
        :
%read signal from my board
signal=query
(my_RP,'ACQ:SOUR1:DATA?');
        :
        :
        :
end
        :
```

Fig.5 Redpitaya hardware interface

This is the main reason that reditaya stem lab is much attractive in the field of signal processing. Various prewritten modules are there in red-Pitaya that helps signal processing much easier. With the help of inbuild hardware description converter [HDL Converters] in redpitaya helps to convert various programs in various platforms.



\

Fig.6 Software infrastructure

Pyrpl is terminal in that we can run bit files of various programming languages like Verilog, python, mat code, c, etc. The bit file generated by the various programming languages is transferring through an ethernet cable to flash FPGA. This data will be sent for various digital signal processing[DSP] in DSP blocks. Red pitaya consists of programmable systems and programmable logic and client pc. We can control entire repitaya including programmable system and programmable logic with the help of client pc [python interpreter]. The programmable logic block is directly receiving and transferring the input and output analog data. The programmable system is the intermediate between python interpreter and programmable logic block. The programming languages used are, *Verilog*: Verilog is a hardware description language(HDL) that is used for designing the digital circuits. We are designing the FPGA blocks for our specific purpose with the help of the Xilinx ise design tool... The programming language required for Xilinx ise design is either VHDL or VERILOG. Here we are using Verilog for the design purpose. ISE is software invented by Xilinx for synthesizing and verifying 'the digital circuit that we have programmed with the help of hardware description language(HDL). *Matlab*: "MATLAB is a high-performance language for technical computing". It integrates computation, visualization, and programming in an easy-to-**use** environment where problems .Matlab software is invented by MathWorks. These programming platforms can integrate with the help of redpitaya. For that, we need to generate a bit file and that bit file needs to dump into fpga with the help of pyrpl and transfer the generated bit file with the help of ethernet cable. Here ethernet cable is using for the internet as well as for controlling and transferring data into the redpitaya. Board will act as the internet of things platform while connecting the internet and we can control devices using the board.

IV. DESIGN AND IMPLEMENTATION

Hardware design of a blind source separation system consists of Transducer, ADC, Memory, FPGA processor, DAC, CRO/DSO. The input signal received from various(unknown) sources by the transducer will be in the form of mixed-signal. The received input audio signals by the transducer will be sent to ADC to convert into digital form. In general audio signals are analog and for digital signal processing need to convert into digital form with the help of ADC(AD9648).AD9648 is a dual-channel ADC of resolution 14bits. The sampling rate of AD9648 is 125 MSPS/105 MSPs. A single analog power supply of 1.8v is required for operating this ADC."The standard serial port interface(SPI) supports various product features and functions, such as data output formatting, internal clock divider, power-down, DCO/data timing, and offset adjustments". The digital output pins of ADC will contain the digital data and storing digital data in FPGAs memory for signal processing purposes. One of the important requirements to implement Blind source separation in a hardware [FPGA] is its memory availability and capability of memory designing according to our needs. Memory present in zinq7010 FPGA is DDR3 and DDR2 SDRAM[Double Data Rate 3/2Synchronous Dynamic Random-Access Memory. The bandwidth of these memories is high[72 bits, 64 bits]. The attractive features of DDR3 are 8-bank support, 8-word burst support, ODT support(On-die termination), ECC support, Clock ratio for memory to FPGA interface is 4:1 and 2:1, CAS write latency will support 5-10 cycles, support Support UDIMM, RDIMM, SODIMM of sing and dual-rank, etc. Preprocessing is a step to reduce the complexity of the mixed signal. For that need to design multiple memories according to our requirement and need to use a few of it simultaneously. This is one of the reasons for choosing FPGAs for signal processing.

The next section is about the blind source separation[BSS] algorithm. FASTica[FAST independent component anayisis] is using as a BSS algorithm. It has two-part, one is a pre-processing algorithm and the second is the “kurtosis” algorithm. In the pre-processing algorithm, we are simplifying the complex input data and this simplified data is giving to the kurtosis algorithm. The kurtosis algorithm will separate the mixed simplified signal after preprocessing by finding a linear transformation that can maximize the mutual independence of the mixture. FASTica algorithm is the combination of both the preprocessing algorithm and the Kurtosis algorithm. After signal processing[BSS] done by FPGA, the separated source signal will move to FPGAs[master] output. FPGAs output pins are connected to DACs input pins where after digital signal processing again we are converting the digital signal to an analog signal. The output pins of DAC is connecting to CRO[Cathode ray oscilloscope]. Through CRO we can analyze the separated input signals and compare the correlation of different input signals for knowing the accuracy and efficiency of the system. The second method is the usage of DSO[Digital signal oscilloscope], without using DAC we can connect the output of FPGA directly to the DSO and we can analyze the digital values. The comparison between the results of different BSS algorithm give FASTica is better in performance as well as speed.

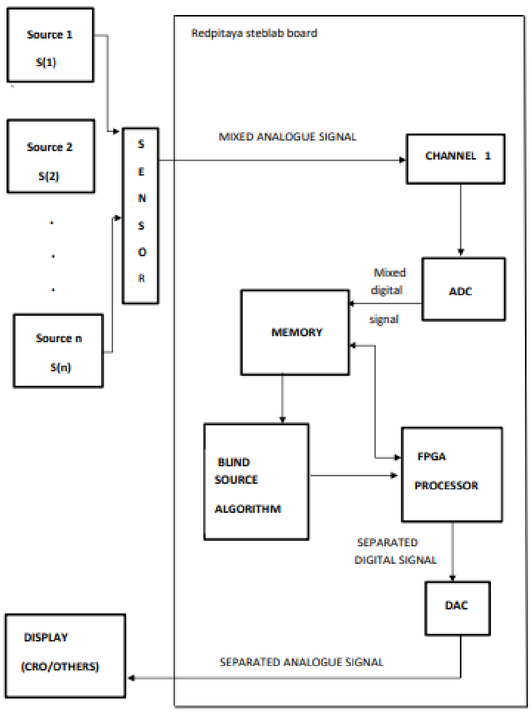


Fig.7- Hardware design

Transducer/ Sensor: A transducer is a kind of sensor that covert some form one form of energy to another [eg: audio to electrical]. Here microphone will act as a transducer. Tranducer[microphone] will capture the audio signal and convert it into an electrical form. Audio signals are analog nature, the audio signal from the microphone will receive by an ADC to convert it into a digital signal for signal processing.

ADC[Analog to digital converter]- AD9648: AD9648 is a dual-channel ADC of resolution 14bits. The sampling rate of AD9648 is 125 MSPS/105 MSPs. Total four data input channels are there for two ADCs[Duel channel]

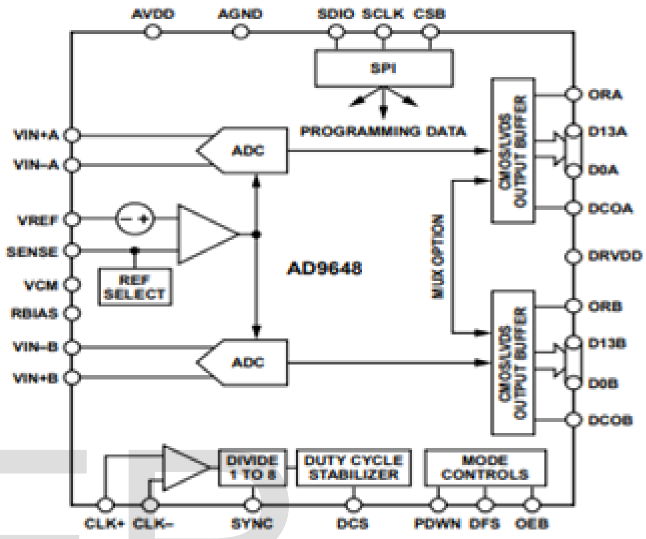


Fig-8 ADC Functional block diagram.

SPI Protocol: Here we are using serial peripheral interface technique for data communication. SPI generally has a data bus and clock. According to the clock cycle, only each bit will be sent. The clock is acting like an oscillating signal that informs when to transfer and sample each bit on the data bus line to the receiver. SCLK the serial clock shift pin acts like a synchronize serial interface reads and writes. Serial input and output pin [SDIO] has two purposes. First is typical serves as input or output depending on instruction being received. Second is the relative position in the timing frame. The third and final pin CSB[Chip select bar] which controls the read and write cycle by keeping active low or high. The falling edge of chip select bar and the rising edge of the serial clock indicating framing that means Start transfer data bits. If CSB is in an active low condition that means the device is in permanently enabling stage called streaming. “During an instruction phase, a 16-bit instruction is transmitted. Data follows the instruction phase, and its length is determined by the W0 and W1 bits”. More than this the instruction phase indicates the serial frame is read or write operation allowing the serial port to be used both to program the chip and to read the contents of the on-chip memory. The first bit of the first byte in a multibyte serial data transfer frame indicates whether a read command or a write command is issued . This is the logic behind SPI communication and we have implemented the same protocol. The hardware description language called Verilog is used for programming ADC. First, we initialized the input of ADC and arrange it as register or wire[input/outout]. Then in the positive edge of the clock that is chip select bar is low and SDIO [Serial data input output] then bits start to transmit is one condition. The second condition is Chip select bar is high and SDIO is low at that time the transfer od bit will not happen. The output of ADC is connected to the input of the FPGA processor. Once the converted digital 14-bit data received in the input pins of

FPGA. Then the next step called preprocessing[Simplification of input mixed audio signal].

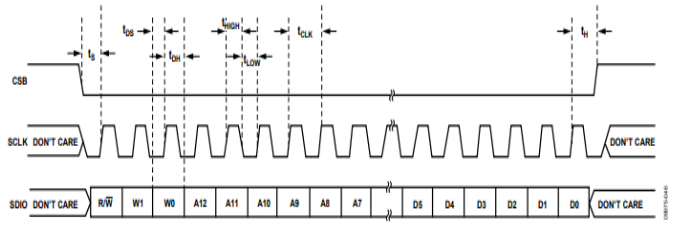


Fig-9 SPI timing diagram.

Memory: ZINQ 7010 FPAGA consists of two types of memory that are DDR3[Double Data Rate 3] and DDR2[Double Data Rate 2] SDRAMs. One of the important requirements to implement a blind source separation algorithm in hardware is memory availability and capability of memory designing according to our needs. In general, a DDR memory essentially regular too but the interface has extra DQS signal. DQS signal is also called a busted clock which only works during reading and writes operation. The extra DQS signal helps to clock the data signals on both the edges[Falling and the rising edge of the clock]. Also, DQS behaves like a clock just for the sampling of data. The address and control signal work in conjunction with the clock.

BACKGROUND OF FASTICA ALGORITHM :

Let us consider a mixed-signal matrix X is,

$$X = FU$$

Where $x = (x_1, x_2, x_3, \dots\dots\dots, x_m)^T$ and the signal arrays are $s = (s_1, s_2, s_3 \dots\dots s_u)^T$ .It contain m observed mixed-signal and m independent unknown sources. Where F is a full rank m by m mixing matrix. The goal of FASTICA algorithm is to separate the unknown source u by estimating mixing matrix D. X and D are expressed as;

$$X = (x1, x2, x3 \dots\dots\dots x(j-1), x(j))$$
$$D = (d1, d2, d3 \dots\dots\dots d(j-1), d(j))$$

Where $j = 1,2 \dots . k$, number of time sample indicate as k.

PRE-PROCESSING FASTICA ALGORITHM :

The preprocessing algorithm is also called a PCA algorithm[principle component analysis algorithm]. In preprocessing we aim to find out the Z matrix. Z matrix is the simplified form of the input signal [x] then we are giving Z matrix directly to the kurtosis algorithm fr signal separation. The main intension of preprocessing is to reduce the complexity of FASTICA algorithm. PCA algorithm, which finds a linear transformation and translates correlative matrix X into Z.

Preprocessing FASTICA afgorithm[PCA] contains two steps :

1. Centering.

2.Whitening.

Centering: Centering is a method in which calculating mean from mixed-signal x and subtract the mean from x. Centering in preprocessing defined as;

$$x(j) = x(j) - E\{x(n)\}$$

where j= 1,2,3…….n. Where n specifies the number of time samples. After the technique centering x becomes a zero[0] mean matrix.

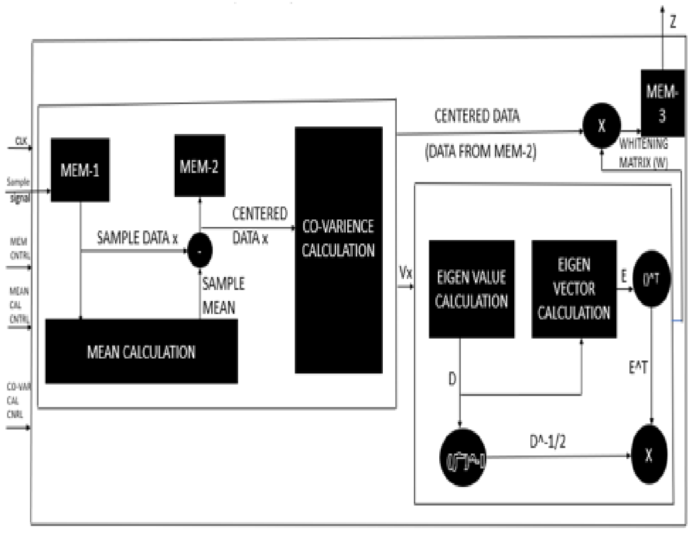


Fig-10 Preprocessing

Whitening: Whitening is the second step of preprocessing. whitening utilizes EVD[eigenvalue decomposition] defined as ;

$$E\{XX^T\} = V_X = EDE^T$$

The covariance matrix of X is V_X, $E = (e_1, e_2, e_3, \dots\dots, e_m)$, is the orthogonal matrix of eigenvectors of V_X. $D = diag(\partial 1, \partial 2 \dots\dots \partial m)$ is the diagonal matrix of eigenvalues of V_X.

The processes of whitening can be described as ;

$$Z = WX$$

Where W is the whitening matrix. The new matrix become white is W, W will be defined as,

$$W = D^{-\frac{1}{2}} * E^T$$

after the whitening process, It is easy to show that elements in the matrix Z are uncorrelated, Because,

$$E\{ZZ^T\} = WE\{XX^T\}W^T = D^{-1/2}E^T EDE^T \ ED^{-1/2} = I$$

After preprocessing, the mixing matrix F transforms into a new one F_0,

$F_0 = WF$, can be received from

$$Z = WX = WFS = F_0 S$$

The new mixing matrix F_0 is orthogonal, can be seen from

$$E\{ZZ^T\} = F_O E\{SS^T\}F_0^T = F_0 F_0^T = I$$

This means that the FASTICA problem of finding the full-rank matrix F is simplified to the estimation of the orthogonal mixing F_0.

Finding Independent sources: The distribution of the sum of independent random variables within close range to a Gaussian, based on the central limit theorem. To find independent components the measurement of gaussianity is used. kurtosis or fourth-order cumulant to calculate non-gaussianity as higher-order statistics. The " **kurtosis"** of a zero-mean random variable K is defined as,

$$kurt(k) = E\{k^4\} - 3(E\{k^2\})^2$$

Where $E\{k^{4}\}$ is the fourth order-moment of k, E$\{k^{2}\}$ is the second-order moment of k. That means kurtosis is 0 [zero] for a gaussian random variable. kurtosis is either positive or negative if k is a non-gaussian random variable. Therefore using the absolute value of kurtosis the non-gaussianity is measured.

[7], [5] proposed a floating-point algorithm for performing the computations needed in FASTICA. The meaning of the floating-point in this case is not the number representation. "It is an iteration scheme to find the local extreme value of the kurtosis for efficiently estimating the non-Gaussian independent components". The normal method of the "FastICA" algorithm follows as,

a) Take an initial random vector $q(0)$ and divide it by its norm. Let $r = 1$.
b) Let $q(r) = E\{[Z[Z^{T}q(r-1)]^{3}\} - 3k(r-1)$.
c) Divide $q(r)$ by its norms.
d) If $|q^{T}(r)q(r-1)|$ is not close enough to 1, Let $r = r + 1$, and go back to step 2, . Otherwise, the algorithm is convergent and outputs $q(r)$.
e) q is the separating vector used to find out the estimated independent component (s_est)
f) The iteration index is r.

If there are" n *-independent components*" to be separated, each converging result of the "FastICA "algorithm "q" is one of the columns in demixing matrix C. Assuming q is a column of C. Then s_est is defined as,

$$s\_est = q^{T}Z$$
$$s\_est = (s\_est_{1,}s\_est_{2}, s\_est_{3} \ldots\ldots\ldots s\_est_{n})^{T}$$

To make sure that each estimation s_est is a separate independent component. The FastICA algorithm joins the following orthogonal projection:

$$q = q - \overline{CC}^{T}q$$
$$q = q/||q||$$

Where $\overline{C}$ is a matrix whose columns are the previously determined columns of C.

Optimization of FASTICA algorithm :

In general independent component analysis[ICA] algorithm separates the source signal from mixed-signal by finding a linear transformation that can maximize the mutual independence of the mixture. For this need to develop a separating matrix W. This method is difficult to sort out the gradient information of the independent sources. So instead of conventional gradient descent method, introducing a new genetical algorithm for the optimization of algorithm.

DAC [Digital to analog converter]

DAC is a device that converts the digital signal into an analog signal. In this case, after Signal processing is done by FPGA processor the separated source signal will be in the form of binary that is digital. For analyzing the waveform in Cathode ray oscilloscope [CRO] for the comparison purpose need to convert the digital signal again back to analog. The processed data in FPGA will gen in FPGAs output pin. We are wiring internally the pins of FPGAs out and DACs input. DAC again resample the processed FPGA signal and it will convert back into an analog signal. This analog signal; can analyze in a CRO.

DISPLAY[CRO]

Cathode ray oscilloscope is an electronic test device. When different input signals are given, they can view it as a waveform in CRO and also it helps to analyze and study the waveform.

IV. RESULTS

Fig-11 Memory Design[FPGA]

Here we designed memory according to our requirement and created memory has tested in Zinq 7000 FPGA. DDR2/DDR3 is used for storing large amounts of data for efficiently performing various algorithms. There are Dedicated blocks of memory, we have Designed five the memory according to our input data requirement (created memory= 64*8 (5)).

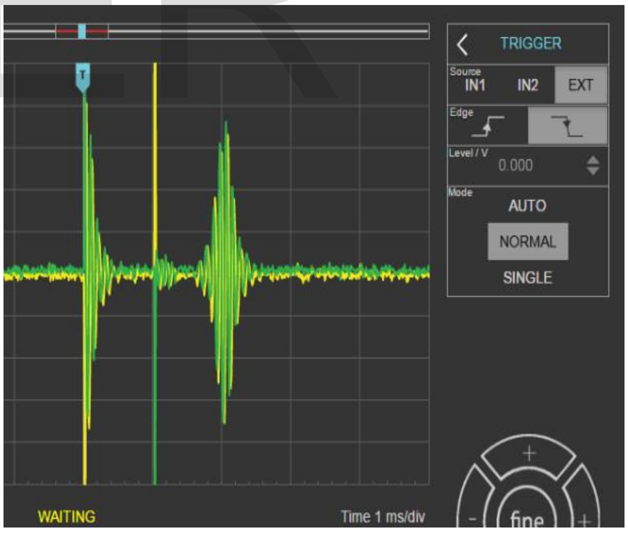


Fig-12 ADC output

AD9648 is a dual-channel ADC of resolution 14bits. The sampling rate of AD9648 is 125 MSPS/105 MSPs. The input signal which we have given as sampled by the ADC at the rate of 125 MSPS. Fig-12 is the output waveform of ADC and we have tested in redpitays inbuilt CRO. Red pitraya board can access all signal analyzing techniques in their online application platform.

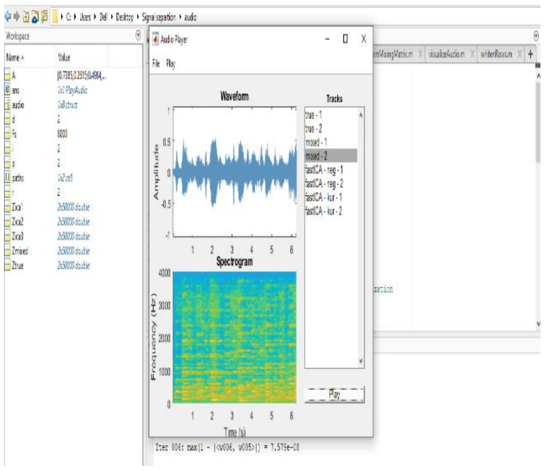


Fig- 13 optimized FASTica output

The output of optimized hardware output has tested in redpitaya stemlabs through its inbuilt Matlab application. The math code we wrote in Matlab will be converted into Hardware description language with the help of an inbuilt HDL converter. We compared various output parameters with recent research papers of the Hardware implementation of FASTica algorithm and our proposed Implementation of FASTica algorithm using FPGA showing better speed and less output correlation.

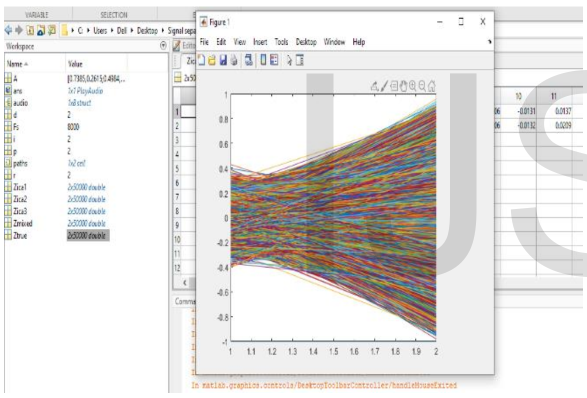


Fig-14 The contour representation of uncorrelated signals

V. CONCLUSION

Implemented the Blind source separation algorithm[optimized FASTICA] using FPGA. The optimized[grnitic algorithm] fASTica algorithm accomplished with ADC and DAC peripherals for real-time signal processing. The implementation of the FPGA (zinq7000) based optimized FASTICA algorithm makes real-time blind source signal separation possible. The proposed FPGA based architecture helps high-speed signal processing of optimized FASTICA with a high sampling rate up to 125 MS/s by hand-coding the optimized FASTICA algorithm in Hardware description language Verilog and high-performance language for technical computing like MATLAB. Usually, the output correlation of input signals is high in blind source algorithm, our proposed work overcome the scenario with the help of an optimized FASTICA algorithm[genetic algorithm]in high-speed FPGA called Zinq7000 in red pitaya. The simulation of the algorithm and the hardware experiment demonstrate the proposed implementation of a blind source separation algorithm [Optimized algorithm] is capable of real-time signal processing such as fetal heart rate monitoring, radar noise cancellation, and speech signal enhancement.

REFERENCES

[1] Chang-Min Kim, Hyung-Min Park, Taesu Kim, Yoon-Kyung Choi, and Soo-Young Lee, Member "FPGA Implementation of ICA Algorithm for BlindSignal Separation and Adaptive Noise Canceling", "IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 14, NO. 5, SEPTEMBER 2003"
[2] Charuyuphan Chareonsak, Farook Sutta, Yu Wei, and Xiong Bing "DESIGN OF FPGA HARDWARE FOR A REAL-TIME BLIND SOURCE SEPARATION OF FETAL ECG SIGNALS" "2004 IEEE International Workshop on Biomedical Circuits & Systems."
[3] Geethu R. S, Sudhish N. George, Krishna Kumar M "A Proposal for Source Separation of Heartbeat Sounds and its FPGA", "2012 International Conference on Communication Systems and Network Technologies."
[4] Yongsoon Lee, Seok-Bum Ko, "FPGA IMPLEMENTATION OF A FACE DETECTOR USING NEURAL NETWORKS" "IEEE CCECE/CCGEI, Ottawa, May 2006"
[5] N. Shirazi, A. Walters, and P. Athanas, "Quantitative analysis of floating-point arithmetic on FPGA based custom computing machines," "in Proc. IEEE Symp. FPGAs Custom Comput. Mach., 1995, pp. 155–162"
[6] L. H. Arnaldi "Implementation of an AXI-compliant lock-in amplifier on the RedPitaya open-source instrument" "2017 Eight Argentine Symposium and Conference on Embedded Systems (CASE)"
[7] 7] M. Kumm, H. Klingbeil, and P. Zipf, "An FPGA-Based Linear All-Digital Phase-Locked Loop," "IEEE Trans. Circuits Syst. I Regul. Pap., vol. 57, no. 9, pp. 2487–2497, sep 2010. [Online]".