International Journal of Scientific & Engineering Research Volume 11, Issue 8, August-2020
ISSN 2229-5518

SENSOR DATA ACQUISITION AND DE-NOISING USING FPGA

Harikrishnan.K, Vishwas.H.N, Vineetha Jain K.V, Dr.Ramesh Chinthala(Dept of ECE)

Department of Computer Science & Engineering, Amrita School of Engineering, Bengaluru,
Amrita Vishwa Vidyapeetham, India
hari.krishnalayam@gmail.com, hn_vishwas@blr.amrita.edu, jain_vineetha@blr.amrita.edu,
c_ramesh@blr.amrita.edu

Abstract-As the electronics industry and user needs are developing rapidly, the design of embedded systems should match their standards. Every equipment we encounter with our day-to-day life is getting automated, the world is getting smarter day by day. As the usage of sensors is increasing, the processor should be capable of handling an enormous amount of data. Our work will focus on designing and implementing an FPGA (Field Programmable Gate Array) based DAQ (Data Acquisition) system. Sensors and FPGA both are independent research areas, their combination is an emerging research area. The prime reason to choose FPGA over other MCUs (Micro-controller units) is the top processing speed. The system mainly includes sensors, a DAQ device, and signal processing (Denoising). Noise affects real-time systems. Noise is random and unpredictable, so noise reduction processing is necessary to get an accurate measurement. Denoising algorithms are very complex, so FPGAs are better than MCUs. The proposed system fetches the real-time signals from sensors, performs the digitization, and the digitized data can process accordingly. We observe it that only a few works are available for real-time applications.

Keywords- DAQ, FPGA, de-noising, FFT, ADC, DAC, Temperature sensor

I. INTRODUCTION

Measuring an electrical or physical phenomenon such as voltage, pressure, current, sound, or temperature with a computer known as data acquisition. Frequency synchronization of the input signal and the processor is the major considerable factor, else it will cause a mismatch in signals and calculations. The operating frequency of the external sensor should match with the processor you are using. A real-time system needs an application to meet the average deadlines of a set time with minimal variability while processing an external event. Time deadlines for critical real-time systems are very strict, it must meet these deadlines in every case. An example of such a case is when a controlled radiation dose has to be delivered to a sample. It bases most of the sensor designs on the MCU platform, many micro controllers design typically mix multiple interface methods. In a very simplistic form, we can view a micro-controller system as a system that reads from inputs, performs processing, and writes to output. For real-time applications and complex computations, we cannot depend on MCUs because of their poor execution time and less energy efficiency. DSPs (Digital signal processor) cannot match both improved performance and low energy consumption requirements of sensors.

FPGAs can solve this issue. An FPGA is an integrated circuit (IC) which can program for different algorithms after fabrication . Modern FPGA devices comprise up to 2 million logic cells that can be configured to implement a variety of software algorithm. Although the traditional design flow is more similar to a regular IC than a processor. It provides significant cost advantages compared to an IC development effort and offer the same level of performance in most cases. Ability to dynamically reconf igured is it‟s another advantage.

Systems based on FPGAs offer many advantages over conventional implementations. Using FPGAs makes you independent of component manufactures, because the functionality is not in the module itself but in the configuration. As the largest user group is to be addressed with standard components, these are usually a compromise between performance and compatibility. FPGAs offer the possibility of developing systems that are exactly tailored to the intended task which thus works efficiently. Complex tasks are often solved by software implementations with fast processors. FPGAs are ideal for applications in time-critical systems and offer a cost-

effective alternative, which, via parallelization and adaption to the application, provide a significant speed advantage compared to processor-based solution. In contrast to a software-based solution with real-time operating systems, and actually, deterministic behavior can be achieved using FPGAs. Because of the offered flexibility, we can carry even complicated calculations out in a quick time. The amount of data to be processed in current systems are already increasing, so they can no longer be processed promptly using sequentially operating systems. Using FPGAs offers an outstandingly scalable solution via parallel processing of data.

The FPGA device includes embedded memory elements that can be used as random-access memory (RAM), read only memory (ROM) or shift registers. These elements are block RAMs (BRAMs), ultra RAM blocks (URAMS), LUTs and shift registers (SRLs). The BRAM is a dual port RAM module instantiated into the FPGA fabric to provide on-chip storage for a relatively extensive set of data. These 2 types of BRAM memories available in a device can hold either 18k or 36k bits. The dual-port nature of these memories allows parallel, same-clock-cycle access to a different location.

FPGAs have high efficiency because of the architectural flexibility, especially data parallelism and re-configurability. This means FPGAs can provide greater performance compared to MCUs and will be more flexible than ASIC (Application-specific integrated circuit).

Noise affects real-time systems. Noise is random and unpredictable, so noise reduction processing is necessary to get an accurate measurement . Denoising algorithms are very complex, so FPGAs are better than MCUs.

II. LITERATURE SURVEY

Data acquisition plays a major role in the sensor based applications and different industrial fields [1] mainly discussing about the significance and importance of collecting data and its processing part. It also compares their proposed method with the currently existing methods, what they proposed was an FPGA based data acquisition system. They are also mentioning some disadvantages of the currently used methods and how FPGA can solve those issues and make the system more efficient. The processing speed of FPGA was the major factor attracted them to replace other MCUs with FPGA. When coming to the testing and results, they haven"t done any implementation some simulation results are there.

When it comes to sensor interface part. Shi-zhen Huang and Rui-Qi Chen[2]talks about the interfacing of sensor with FPGA. They are introducing a new method which is a combination of both RISC architecture and FPGA to make the board acts like an MCU. They are not using any standard FPGA board, the board they have used is Hummingbird E203 and in the title only it is mentioning FPGA based actually most of the processing operation is done with the help of ARM. It only talks about digital sensors. The properties of digital sensors and what all things we should care about interfacing sensor is mentioned.

Agricultural requirements are also getting automated day-by-day when we consider a large area of farmland, we use many sensors for different types of data collection Swarup S. Mathurkar , Rahul B Lanjewar, Nilesh Patel and Rohit S.Somkuvwar focusing mainly[3] mentioning about an agriculture monitoring system. They have used many sensors (temperature, pressure, humidity) and based on the change in sensor readings the system triggers some actions/operations (like turning on/off water pump). Sensors interfaced to the MCU platform not to the FPGA. This is because of the difficulty of interfacing sensors to FPGA , different varieties of sensors are using the interfacing protocol will be different , Analog sensors should connect to ADC. They are connecting the sensors to a Micro-controller and the signals are sending to FPGA through bluetooth. FPGA is only used for the calculation part. Sensor readings are collected by one MCU and sending to FPGA for calculation purpose. MCU acts as a third person in this process.

For the practical implementation of FPGA based systems Shuang Bao, Hairong Yan , Qingping Chi , Zhibo Pang and Yuyin Sun [4] gives a clear idea about everything. Their team has designed one system which detects water pollution. They are clearly mentioning the other fields where research based on FPGA and also they are mentioning that they are providing all this details to provide a reference to others. In the paper they are briefly explaining about the different reconfiguration techniques available in FPGA , which is one of the major advantages of FPGA. Virtex 4 was the board used by them. They are also using one micro controller for sending the data to FPGA and computer but the whole processing is done by FPGA. They had used VHDL code to drive the entire system. The team was mainly focusing on the reconfiguration and they had used one switch matrix(MT8816) for the reconfiguration. They had implemented the system in a small factory and analysed the performance. There were so many supporting files mentioned in the paper I have gone through some of the paper which was useful for my work. Their work mainly focusing in the processing part. FPGA process sensor signals and based on that they are triggering some actions. They have a combination of digital and analog sensors, digital sensors are directly connected to FPGA and analog

sensors are connected to an MCU platform which collects the data and send it to FPGA for further processing. They have mentioned the concepts of memory creation, parallel processing, clock division. etc.

FPGA should process the digital data so the usage of devices like ADC and DAC are required. Satyaki Mascharak and Arghyapriya choudhuri [5] they gives a clear idea about how to access the on-board ADC and DAC, the types of IC"s available in the market, advantages disadvantages, various interfacing protocols. They used spi interface, The on-board ADC as well as DAC is basically a plug and play. The FPGA itself will take care of all the control signals. The usage of the mater clock is the only thing we have to consider. They provide details of control signal usage. There will be only one ADC, if we want to connect multiple sensors we need to use an external ADC. They also mentions about the in-built ADC may not cater every user needs. They considered the implementation of external ADC as a future work. The board they used was spartan 3.

Digital signal processing is one of the most rapidly developing subjects over the past decades and nowadays it has been widely used in the fields of communication, sonar, radar, image processing. In general, hardware implementation of digital signal processing algorithm can be accomplished in three ways which are general-purpose programmable DSP, specific function DSPs and ASIC, user programmable FPGA. Wang Yaquin, Liu Xuebin, Hu Bingliang [6] says that field Programmable Gate Arrays have become popular platform for digital signal processing, however, the conventional method, which is essentially writing a source code in a hardware description language to design a product on a FPGA, is time consuming and complicated. This restricts the implementation of complex signal processing algorithm on FPGA.In this paper, two fast design methods of algorithm implementation on FPGA were proposed-ExtremeDSP solution and Smart IP-Core technology. The AccelDSP Synthesis tool, as the main component of ExtremeDSP solution allows transferring a MATLAB floating point design into a hardware module that can be implemented on a Xilinx FPGA. Employing predefined cores can cut the design time and significantly reduce risk while having access to the best performing and lowest cost component available. Implementation of a LMS-adaptive filter employing these two methods has been achieved and the simulation results indicate that both of the two fast designs have high performance.

De-noising plays a major role in signal processing. Nannan Zhang, Zedong Nie, Yu Luo, Leilei Du, Xiaohui Wang, and Lei Wang [7] discuss about various noises affected by a signal and the efficient de-noising method. They are doing this for an ECG signal.Dynamic Electrocardiograph (ECG) monitoring (known as Holter) plays an important role in the earlier detection and diagnosis of various cardiovascular diseases. ECG signals obtained from Holter systems normally contain a lot of noises and artifacts. These noises degrade signal quality, which may be critical for routine monitoring and diagnosis. To solve the problem, a reconfigurable overlapping fast Fourier transform/ inverse fast Fourier transform (FFT/IFFT) filter for suppressing the power-line interference and the high-frequency noise is presented in this paper. The filter is based on a 12-lead Holter system with a high-performance analogue front-end and a field-programmable gate array (FPGA) for enhanced digital processing. This paper analyzes the performance of the reconfigurable overlapping FFT/IFFT filter in ECG de-noising applications and validate it by real-world emulations. Furthermore, the de-noising performance of the reconfigurable overlapping FFT filter was evaluated.

The references and their supporting papers gives a clear idea about how the combination of sensors and FPGA is becoming an emerging research area, and their importance. Different Denoising methods and difficulties during implementation is mentioned in reference work. It gives a basic knowledge of FPGA boards and different protocols, difficulties during hardware implementation and things to consider while connecting sensors to different processors. The benefits of FPGA over other MCUs is also clearly understood.

III. FPGA BASED DATA ACQUISITION SYSTEM

HIGH LEVEL DESIGN

Our system mainly comprises sensor, a processor which collects the data from the sensors and performs the signal processing and a CRO for displaying the result. We use the sensor to measure physical quantities from the external world. We use ADC and DAC to perform the conversion.

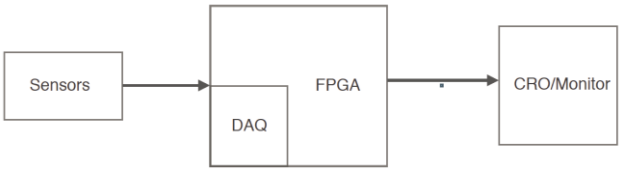

Fig.1- Overall block diagram

LOW LEVEL DESIGN

Temperature is among the most frequently measured analog parameters. We might expect this since temperature affects most electrical, chemical, mechanical, and environmental systems directly or use its value to control other relevant processes. So we use a temperature sensor. The output of the sensor will be an analog value, which FPGA can not take for further processing before conversion. ADC is used to convert the analog signal to digital. FPGA takes this digitised signal as the input for further processing.

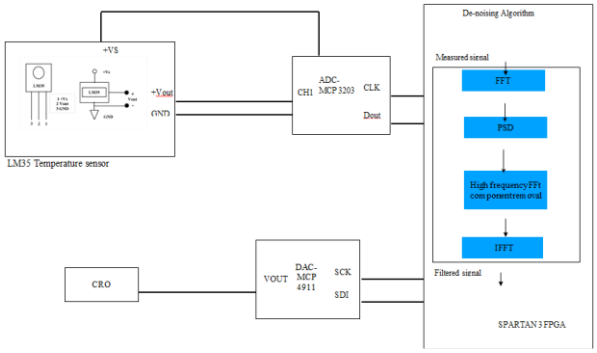

Fig.2- Hardware implementation of the proposed system

We can divide the design into 3 parts

1. ADC part
2. Denoising
3. DAC part

1. **ADC.**

We use MCP3202, for that we wrote a code in verilog. The interface protocol between FPGA and ADC is SPI. The Spartan3 FPGA has 2-channel 12 Bit SPI ADC. As you know in synchronous serial communication, there is a clock line (SCK in case of SPI) which synchronizes the transfer. The clock is always controlled by the MASTER. In our case the FPGA is the MASTER, and the MCP3202 is a slave on the bus. SPI is full duplex, which means data can be sent and received simultaneously.

An SPI transfer is initiated by the MASTER pulling the CS line low. The CS line sits at HIGH during idle state. Now master can write to the bus in 8bit (or 1 byte) chunks.

One most important thing to note about SPI is that for every byte MASTER writes to SLAVE, the MASTER receives one byte in return. So the only transaction possible is the exchange of data.For SPI, we have to take care about 4 signals.

CS - chip select for powering on and powering off. The conversion will happen at an active low of cs (CS=0).

CLK - This is very important, FPGA"s master clock will work on a very high frequency say approximately 50 Mhz, ADC don"t need that much clock frequency ADC need only Khz or 1Mhz range of clock frequency to work. So we need to write a clock divider code based on what frequency we want.**MISO** - Master in slave out, Transmits data from slave to master

MOSI transmits data from master to slave

We give input to the Channel 1.

We can collect the output from the Dout pin of the ADC. There is another pin called Din, which is a channel select. While collecting the ADC externally we have to give proper VCC and ground

All inputs and outputs wrt vss = -0.6 to0.6v

ESD protection on all pins = >=4kv.

Fclk = 18*fsample

Specified temperature range = -45 to 85

Maximum active current at 5v = 550micro A

Clock frequency = 1.8Mhz(5v)

0.9Mhz (2.7v)

POWER REQUIREMENTS

Operating voltage = 2.7 to 5.5 v

Operating current = 550microA (VDD = 5v , Dout unloaded)

PIN FUNCTION TABLE

cs^-/SHDN = Chip select / shutdown input

CH0= channel 0 analog input

CH1 = channel 1 analog input

VSS = Ground

DIN = serial Data In

Dout = Serial Data out

CLK = serial clock

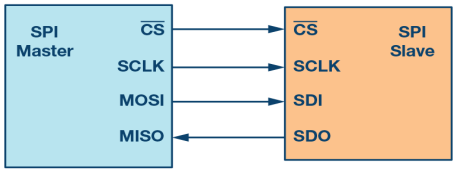

Fig.3- Representation of SPI interface

2. **Denoising**

Denoising is removing noise from a signal. Noise reduction techniques exist for audio and images. Noise reduction algorithms alter signals to a greater or lesser degree.

All signal processing devices, both analog and digital, have traits that make them susceptible to noise. Noise can be random or white noise with an even frequency distribution, or frequency dependent noise introduced by a device‟s mechanism or signal processing algorithm.

In electronic recording devices, a major noise is created by random electron motion because of thermal agitation at all temperatures above absolute zero. These agitated electrons rapidly add and subtract from the voltage of the output signal and thus create detectable noise.

Common sources of Interference

- On-board clock
- Harmonic distortion
- Intermodulation distortion
- Power supply switching
- poor decoupling

There is a beautiful property of mathematics, where every signal can be thought of as the sum of a bunch of sine waves. So real world signal has the same property, they might look like a tangle in a time domain then a nice, neat shape once we have done FFT. The FFT helps you see what kinds of signals are present in your system. Specifically, it breaks down your complicated signal into separate sine waves. The voltage changing sinusoidal over time will represent in the FFT by one spike because of one frequency. The figure below shows the signal and FFT.

We can use FFT to de-noise the signal.It is possible to compute the fast Fourier transform of this noisy signal using the fft module. The power spectral density (PSD) is the normalized squared magnitude of f, and shows how much power the signal contains in each frequency. The noisy signal will comprise over one peak. It is possible to zero out components that have power below a threshold to remove noise from the signal. After inverse transforming the filtered signal, we find the clean and filtered time-series match well. Specific commands are there to access FFT modules.

FPGAs have FFT modules in it. We call those kinds of modules in FPGA as IP cores. Intellectual property cores are standalone modules that can use in any FPGA. It is a part of the growing electronic design automation (EDA) industry. We can categorise it as hard IP cores and soft IP cores.

Hard IP cores- These are the part of the FPGA-independent modules. We have to configure the location and provide interface connectivity with other modules, clock and reset.

Soft IP cores- These are flexible and do not depend on vendor technology.

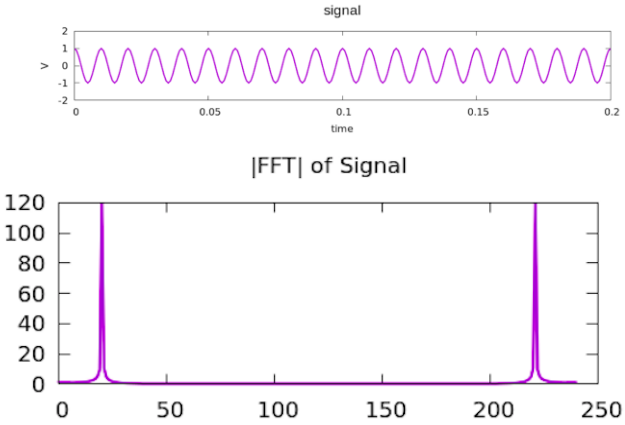

Fig.4- Signal (Top) and its frequency (Bottom) domain representation

1. **DAC**

We use MCP4911, for that we wrote a code in verilog. The interface protocol between FPGA and ADC is SPI.

For SPI, we have to take care about 4 signals, same as that of ADC.Here the FPGA connected to DAC with only three pin which is CS, SCK, SDI. Those signals are control signal for DAC. We send the data bit to SDI of DAC and CS for start the conversion. We connect VDD and VREF to power supply (+5v) with decoupling capacitor. We connect LDAC and VSS to ground. We connect the PCB connector to DAC output on which we can connect the multimeter or CRO to measure the DAC output.

Supply Voltage Pins (VDD, VSS)

VDD is the positive supply voltage input pin. The input supply voltage is relative to VSS and can range from 2.7V to 5.5V. The power supply at the VDD pin should be as clean as possible for excellent DAC performance. We recommend it to use an appropriate bypass capacitor of about 0.1 µF (ceramic) to ground. An additional 10 µF capacitor (tantalum) in parallel recommends to further attenuate high-frequency noise present in application boards. VSS is the analog ground pin and the current return path of the device. The user must connect the VSS pin to a ground plane through a low-impedance connection. If an analog ground path is available in the application Printed Circuit Board (PCB), we highly recommend it that the VSS pin be tied to the analog ground path or isolated within an analog ground plane of the circuit board.

Pin function Table

Chip Select (CS^-)

CS^- is the chip select input, which requires an active-low signal to enable serial clock and data functions.

Serial Clock Input (SCK)

SCK is the SPI compatible serial clock input.

Serial Data Input (SDI)

SDI is the SPI compatible serial data input.

Latch DAC Input (LDAC^—)

The LDAC^— (latch DAC synchronization input) pin is used to transfer the input latch register to the DAC register (output latches, VOUT). When this pin is low, we update VOUT with input register content. We can tie this pin to low (VSS) if we desire the VOUT update at the rising edge of the CS^ — pin. An external control device can drive this pin such as an MCU I/o pin.

Analog Output (VOUT)

VOUT is the DAC analog output pin. The DAC output has an output amplifier. The full-scale range of the DAC output is from VSS to G*VREF, where G is the gain selection option (1x or 2x). The DAC analog output cannot go higher than the supply voltage (VDD).

Voltage Reference Input (VREF)

VREF is the voltage reference input for the device. We use the reference on this pin to set the reference voltage on the string DAC. The input voltage can range from VSS to VDD. We can tie this pin to VDD.

FPGA parallelism versus Processor architecture

When compared with processor architectures, the structure that comprise the FPGA fabric enables high parallelism in application execution. The custom processing architecture generated by the Vivado HLS compiler for a software program presents a different execution paradigm, which we should take into account when deciding to port an application from a processor to an FPGA.

The FPGA is an inherently parallel processing fabric capable of implementing any logical and arithmetic function that can run on a processor. The primary difference is that the Vivado HLS compiler, which is used to transform software description into RTL, is not hindered by the restrictions of a cache and a unified memory space. The computation of Z is compiled by vivado HLS into several LUTs required to achieve the size of the output operand. As a general rule, 1 LU equivalent to 1 bit of computation.

IV. RESULTS

Every FPGA will have an on-board ADC which may not cater to the high frequency user needs. So we implement external ADC setup which can use for any kind of application. For verifying the result, we took input from the temperature sensor and convert it to digital values and displayed in the LCD screen. The on-board ADC is basically a plug and play. The external interface setup will be same for any ADC which helps to work with different varieties of converters, Changing the IC or using different manufactures will not be an issue anymore.

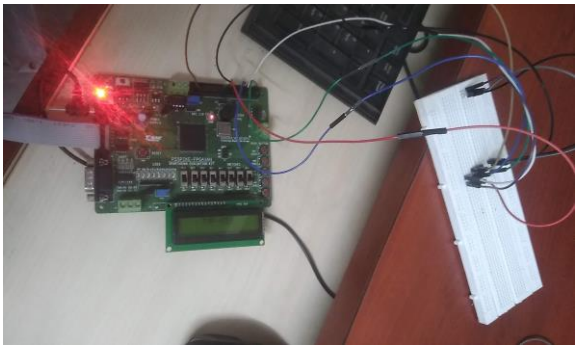

Fig.5- External interface of temperature sensor and ADC to FPGA

The FPGA should process only digital value. So we require DAC. We verified the output of DAC by converting the sensor data back to analog, but the amount of noise affected by the signal was more than that we expected. For testing the DAC itself, we fed digital values one clear sine wave to the DAC input and verified the output by connecting a CRO, to make sure that the DAC works perfectly. In-order to reconstruct the pure input signal from the sensor denoising is a must, else whatever operations we perform on the signal will get affected on the noise.

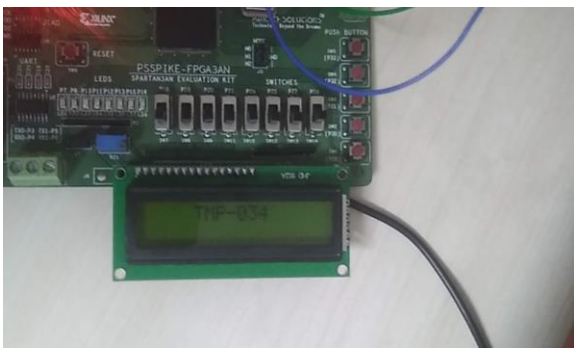

Fig.6-.LCD showing temperature value

We can use FFT algorithm for de-noising. It is possible to compute the fast Fourier transform of this noisy signal using the fft command. The power spectral density (PSD) is the normalized squared magnitude of f, and shows how much power the signal contains in each frequency. FPGAs has FFT in-built modules.

Fig.7- Porting digital values of a triangular wave through FPGA(Left)and DAC output(Right)

We verify the result using MATLAB simulation.
We will consider a function of time f(t)-
f(t) = sin(2πf1t) + sin (2πf2t)

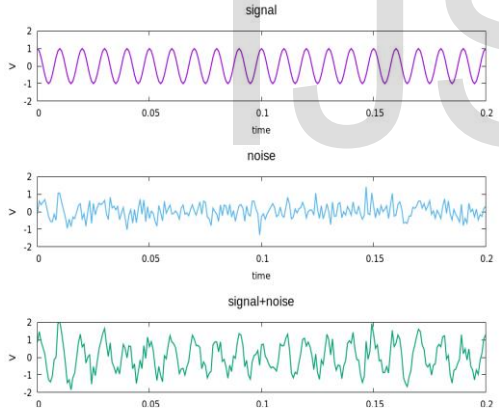

Fig.8- Representation of a signal (Top), noise (Middle), signal+noise (Bottom) in time domain

With frequencies f1 = 50 and f2 = 120. We then add a sizeable amount of Gaussian white noise to this signal.
Then we will compute the FFT- f^, the FFT is basically a one line command. What we get will be a vector Fourier coefficients each of these will be a complex valued entry, it has a magnitude and a phase. The magnitude tells you how important that frequency and phase tells you if it is more cosine or more sine. then we will calculate the Power spectral density (PSD) depending on the input data the PSD have units of power that"s why it"s called the power spectral density. Last thing we have to do is to create a frequency vector we have to figure out each

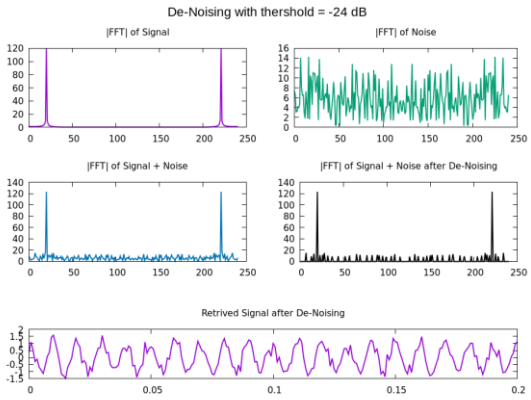

Fig.9- Representation of FFT of signal and noise (Top), signal + noise and noise after denoising (Middle)in frequency domain, Retrieved signal after denoising (Bottom) in time domain. With denoising threshold -24dB

element corresponds to a specific frequency from low frequency to high frequency and we have to find out our fundamental vector of frequencies, from lowest frequency 0 to the highest frequency n. while ploting the X- axis will be in units of frequency and y-axis will be in a unit of power, will tell us which frequencies has more power. Even if the signal is super noisy when we compute the power spectrum after doing FFT we can see the signal as a very large peak and noise like
noise floor. This indicates how to filter out. It is possible to zero out components that have power below a threshold to remove noise from the signal.After inverse transforming the filtered signal, we find the clean and filtered time-series match quite well.

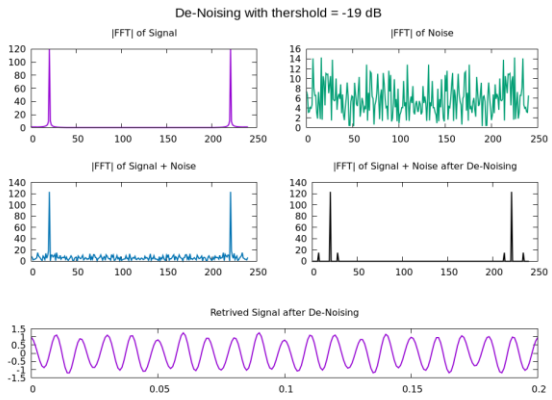

Fig.10- Representation of FFT of signal and noise (Top), signal + noise and noise after denoising (Middle)in frequency domain, Retrieved signal after denoising (Bottom) in time domain. With denoising threshold -19dB

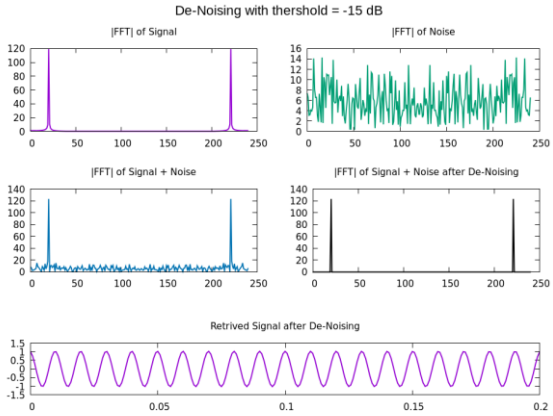

Fig.11- Representation of FFT of signal and noise (Top), signal + noise and noise after denoising (Middle)in frequency domain, Retrieved signal after denoising (Bottom) in time domain. With denoising threshold -15dB

V. CONCLUSION AND FUTURE WORK

An FPGA is an integrated circuit (IC) which can program for different algorithms after fabrication. Modern FPGA devices comprise up to 2 million logic cells that can be configured to implement a variety of software algorithm. As the FPGA should process digital data the ADC interface plays a significant role in the combined area of FPGA and sensors, since the major sensor families are analog. We completed designing and Implementation of ADC and verified the results. We noticed that there will be an enormous amount of noise present in the signal. In-order to filter out the noise, implementation of the FFT based denoising will be good, the results of the Matlab simulation shows how efficient this method is.

Implementing the FFT based de-noising on FPGA should consider as a future work. We can do this in one simple step. We have to consider re-sampling/sample rate conversion. The operating frequency of the external sensor using and operational frequency of the FPGA should match, normally FPGA operating frequency will be „n" times the operational frequency of the sensor. Efficient algorithm like Polyphase Filtering techniques should consider for sampling rate conversion for better result, interpolation and decimation with better efficiency. These two operations will help to reconstruct the input signal in a neat and better state. Any further processing on the signal will give better efficiency and chance of calculation mismatch could reduce.

References

1. Ye Fan "FPGA-Based Data Acquisition System" 2011 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC).
2. Shi-zhen Huang and Rui-Qi Chen "FPGA- based IoT Sensor HUB" 2018 International Conference on Sensor Networks and Signal Processing (SNSP).
3. Swarup S. Mathurkar , Rahul B Lanjewar, Nilesh Patel and Rohit S.Somkuvwar "Smart Sensor Based Monitoring System For Agriculture Using Field Programmable Gate Array" 2014 International Conference on Circuit, Power and Computing Technologies [ICCPCT]
4. Shuang Bao, Hairong Yan , Qingping Chi , Zhibo Pang and Yuyin Sun "FPGA-Based Reconfigurable Data Acquisition System For Industrial Sensors" IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, VOL. 13, NO. 4, AUGUST 2017.
5. Satyaki Mascharak and Arghyapriya choudhuri "implementation of the onboard ADC and DAC on spartan 3E FPGA platform" National institute of technology, Rourkela
6. Wang Yaqin, Liu Xuebin, Hu Bingliang "Implementation of a LMS filter on FPGA employing extremeDSP and smart IP-core design "The Tenth International Conference on Electronic Measurement & Instruments.
7. .Nannan Zhang, Zedong Nie, Yu Luo, Leilei Du, Xiaohui Wang, and Lei Wang "A Reconfigurable Overlapping FFT/IFFT Filter for ECG Signal De-noising"2014 IEEE International Symposium on Bioelectronics and Bioinformatics.
8. https-//forums.xilinx.com/t5/Xilinx-Evaluation-Boards/problem-with-ADC-on-spartan3E/td-p/58159
9. http-//ww1.microchip.com/downloads/en/devicedoc/21034d.pdf
10. http-//www.ti.com/lit/ds/symlink/lm35.pdf
11. http-//databookuw.com/databook.pdf