INTERNATIONAL BURCH UNIVERSITY

FACULTY OF ENGINEERING AND NATURAL SCIENCES

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING



IMPLEMENTATION OF DIGITAL FILTERS ON FPGA

MASTER'S THESIS PROPOSAL

ALADIN KAPIĆ

Supervisor

Assist. Prof. Dr. Dejan Jokić

SARAJEVO

September, 2019

# ABSTRACT

The pace of the world of digital communication is increasing at a tremendous rate. Daily, the engineer is requested to compact more data in the same channel bandwidth with closer channel spacing. By shifting from a two-level system to a four-level system, we double the data bandwidth in a bandwidth limited channel at the expense of requiring a double processing time. Since people use filter in a daily basis, hence applies more usage needs better and cheaper solution. The first phase in digital signal processing is determination of exact problem. Usually, signal contains unwanted frequencies so it is more important to detect frequencies in that period to apply exact filter. There is no need to filter 50Hz if there is not any. Also, amplitude of unwanted frequencies is also important. In some cases, filtering of data is not that much necessary. We can handle that small noise and approximate signal without using any filter device. In this chapter, we will discuss filters on daily basis, filters that common people use every day in real life, also with complex filters with high order. Goal is to research common frequencies that need to be filtered, design filters for daily basis usage, test them and reduce energy efficiency. Most of work required to accomplish goal would be done using simulations and calculations in time and complex domain. FPGA, microcontroller and/or server is just an end point of this process. After study, design, simulations, those platforms would just represent tool for testing and giving results.

**Keywords:**

Digital filters, Laplace transform, Fourier transform, FFT, DFT, DTFT, Z-Transform, real time data processing, FPGA, microcontrollers, servers.

# INTRODUCTION

During 1960, digital signal processing experienced explosive growth, when scientists discovered how to simulate analog filters. In that time, hardware was such a problem. Single analog filter weighed over 1 kg and needed to be separately tuned. Another problem occurred when scientists could not confirm quality of filter, or other component because there wasn’t way to detect where problem were.

One of the first attempts to implement digital filters was solving convolution integral. Implementation is very demanding and requires a lot of processing. To process one second of speech, it would have taken more than one hour. After many months spent solving those problems, finally z-transform and recursive differential equation solving gave a solution [2]. Today, filters are everywhere. In our mobile device, in our TV, PC, laptop, in every electronic device. One interesting thing is that, the more devices we have, more noise we have and of course, the more filters we need.

Frequency domain

Every system, every filter is designed using this domain. Transforms as Laplace Transform or Fourier transform and Z-transform allow to easily go from one domain to other, design and test systems. Most common methods and theorems used in simulation and design of not only filters but systems are Nyquist theorem and Bode diagram. Bode plots show the frequency response of a system. There are two Bode plots one for gain (or magnitude) and one for phase. The Nyquist plot combines gain and phase into one plot in the complex plane. It is drawn by plotting the complex gain g(w) for all frequencies w. That is, the plot is a curve in the plane parametrized by ω.

Analog filters

Today, analog filters are used in plurality of cases to avoid aliasing effect by analog to digital converters (ADCs), or digital to analog converters (DACs). In the case of ADCs, the analog filter ensures that the input signal does not contain any component above the Nyquist frequency. In the case of DACs, the analog filter ensures that no dangerous frequency components are created by the limited resolution and sampling rate of the DAC. Of course, those are not the only applications of filters. Engineers often rely on digital filters. They are “cheaper” to produce and more efficient. The most beautiful thing is that, digital filters can be produced just like analog, going from s-domain to z-domain, and getting discrete differential equation.

Digital filters

Filters function by blocking (suppressing) given frequency components in a signal and passing the original signal minus these suppressed components to the output. In contrast to analog filters, digital filters work by performing mathematical operation on the signal [1]. Depending on the application, different filters are used. When processing audio, or generally where high resolution filtering is required, digital filters are used instead of analog filters. Mathematically, there is a way to get ideal digital filter. Over time, engineers and mathematicians have developed different methods to obtain filters with better resolution. Design of frequency-selective filters usually starts with a specification of their frequency response function. The standard types of ideal frequency-selective filter, either pass or eliminate a region of the input spectrum perfectly and they have abrupt (“instantaneous” or “brick wall”) transitions between passbands and stopbands. However, ideal filters cannot be implemented in practice. So, they have to be approximated by practically realizable filters.

More specifically, in practical filters:

(a) the passband responses are not perfectly flat,

(b) the stopband responses cannot completely reject (eliminate) bands of frequencies, and

(c) the transition between passband and stopband regions takes place over a finite transition band.

Increasing order of filter, we also increase time of computation which means we need faster processors to process data. Sometimes, that is not a problem. But, in real time application, every micro second matters. That gives us a little introduction in this thesis objective.

## Field programmable Gate Array (FPGA), CPU and filters

Ordinary, computation of filter has intensive requirements and often require a large amount of data per second, in order to give better results. Hence, the running time in real time systems is important since the process is determined by the order of filter and quality of filtering. Better filters require more data to process. As noted, the field programmable gate array (FPGA) is efficient hardware solution compared to the central processing unit (CPU) or microcontroller device, since it allows the parallel execution of all processes together with flexible architecture which drastically saves execution time.

Nevertheless, the achievement of FPGA high performance may be realized with the efficient logical design and optimization techniques. In essence, FPGA represents two-dimensional arrays composed of logic blocks such as adders, multipliers, multiplexers (MUX), counter, comparators and flip-flops that are an electrically programmable interconnection between logic blocks. Hence, FPGA is so flexible because it gives a user a way to configure the function of each logic block together with interconnections between them. Thus, a simple logic block of the FPGA may be programmed to provide basic transistor functionality or in the opposite way to perform as a microcontroller with a far more complex configuration.

## VHDL programming and C / C++

Logic block density in FPGA depends highly on length and number of wire segment that could be interconnected through electrically programmable switches. This procedure is called FPGA routing. The overall functionality of FPGA components is established using programming language known as hardware description language (HDL) and there are two types of languages, very high description language (VHDL) and Verilog hardware descriptive language. VHDL is general programming language used for the description of digital circuits and with extension to model both concurrent (parallel) and sequential execution flow. Today, software-hardware tools progress rapidly and they allow an application developer to describe and generate FPGA hardware design from the higher-level languages or software tools such as C**/**C++ or MATLAB. The two FPGA manufacturers like Altera and Xilinx are true leaders in the production of sophisticated FPGA boards for educational and industrial purposes.

In the other hand, C / C++ programming, mostly used in microcontrollers has different approach. It is much easier to produce complex systems using this type platform. Main problem with FPGA and VHDL is word length and / or overflow. That means, before we synthetize system, we have to make sure that word length is well predicted. Those are just introduction problems engineers have to deal with. Everything has its own pros and cons. That is exactly what we are going to talk in this thesis, to synthetize, simulate and test filters on different platforms and models.

# THE MASTER THESIS OBJECTIVE

The main aim of this master thesis is to research different types of filters and their usage, different types of platforms (microcontrollers and FPGAs) and filters behavior on them. Going from scratch and basic mathematical principles, through Matlab simulations to the complex systems as high order filters are, and finally test them on different devices as microcontrollers and/or FPGAs. The entire process could be divided into the following sections:

* Research history of filters, problems occurred by the time and their solutions
* Go through mathematical principles, theories and laws to understand basis of filters (Laplace transform, Fourier transform, DFT, DTFT, Z-Transform) and build models based on them
* Design practical filters for “general population” usage – most people available systems, microcontrollers and other devices
* Compare execution time and quality of filters on different platforms (microcontrollers and FPGA devices) and get insight into potential problems
* Test models on SOC devices – combination of FPGA and Microcontroller and also recognize exact problems during implementation

At the end, focus should be on signals with small frequency spectre, as 50Hz signals and removing noise from them. That means, we will use low pass filters, band pass filters, and / or high pass filters.

# LITERATURE REVIEW

Digital signal processing algorithms are powerful tools that provide algorithmic solutions to common problems. For example, digital filters provide several benefits over their analog counterparts. These algorithms are traditionally implemented using dedicated digital signal processing (DSP) chips, FPGAs, or RISC processors. While these solutions are very efficient at their purpose, they only perform one function in the system and can be both expensive and large. Some devices have implemented hardware multipliers to perform faster computations, independently of current application [3] . A microprocessor based DSP or an FPGA is often used to realize digital filters. However, a microcontroller is a low power and low cost complete system-on-chip microprocessor system. By the proper circuit layout and the appropriate software on its program memory, the microcontroller chip could implement various data operations. Minimizing the cost of digital filters using microcontrollers without degrading the performance and speed of the filter requires various study of the theory of DSP and digital filters. Choosing the convenient microcontroller type and writing the appropriate software may attain a cheaper and more flexible digital filter [4].  Filters can be fixed or adaptive. The design of fixed filters is based on prior knowledge of signal and noise. Adaptive filters have ability to adjust their own parameters according to input signal [5]. The elimination of sinusoidal interference corrupting a signal is typically accomplished with a fixed notch filter tuned to the frequency of the interference. A very narrow notch is usually desired in order to filter out the interference without distorting the signal. However, if the interference is unknown, the center of the notch may not fall exactly over the interference. This application of adaptive filtering is called adaptive noise canceling (ANC) and is fully described for broad-band interferences [6]. The advantages of the FPGA approach to digital filter implementation include higher sampling rates than are available from traditional DSP chips, lower costs than an ASIC for moderate volume applications, and more flexibility than the alternate approaches [7].

According to last three paragraphs, there are many advantages and disadvantages using microcontrollers and FPGA devices. To compare speed of computation, energy efficiency, costs and availability, this master thesis will give much more answers to previous questions and hopefully determine best solution for custom problems.

# METODOLOGY

The first step of the research would require detailed theoretical knowledge of Mathematical tools, transformations, analog signal processing, digital signal processing, available technology, etc.  
Simulink allows users to perform simulation and synthesis of filters, plot their responses, generate different order systems, which means, lot of results will be extracted from there. Next step is research of microcontrollers and FPGA devices with their methodologies, principle of work and possibilities.

After deep research, next thing coming is test of models. Ability to test filters require different platforms. Testing would probably be performed on ESP32 board, Altera De0Nano Experimental Board. Esp32 was chosen hence it already has wireless module inside itself, that opens new ability which is real-time testing on outside servers. What is difference between filtering on far away machines and direct processors that runs in system.

# EXPECTED RESULTS

The primary goal of this research consists of the development of sophisticated filter model, using Matlab, frequency analysis of filter responses and testing on suitable platforms such as, FPGA, microcontrollers and servers. The entire process will be described and developed step by step and will be accompanied by flow and block diagram. This implementation process would include different approach for different platforms. Finally, testing filters on platforms using C/C++ and VHDL programming languages, would give us wanted answers.

# REFERENCES

[1] Juri D. Kropotov, in [Quantitative EEG, Event-Related Potentials and Neurotherapy](https://www.sciencedirect.com/book/9780123745125), 2009

[2] Charles M. Rade, IEEE SIGNAL PROCESSING MAGAZINE [47] NOVEMBER 2006

[3] Silicon labs, AN219 – Using Microcontrollers in Digital Signal Processing applications

[4] [Moutaman Mirghani](https://ieeexplore.ieee.org/author/37077345800), Implementation of matched filters using microcontrollers – IEEE, 23 Feb. 2016

[5] Bernard Widrow, Adaptive Noise Cancelling: Principles and Applications, 1975

[6] JOHN R. Glover, JR , Adaptive Noise Canceling Applied to Sinusoidal Interferences., 1977

[7] Chi-Jui Chou, Satish Mohanakrishnan, Joseph B. Evans, FPGA implementation of digital filters, Telecommunications & Information Sciences Laboratory Department of Electrical & Computer Engineering University of Kansas Lawrence, KS 66045-2228