International Journal of Computational and Electronics Aspects in Engineering
Vol. 1, Issue 1, Dec 2014, Revised in March 2020, pp 1-4.
https://doi.org/10.26706/ijceae.1.1.20141201

Design and Implementation of Equiripple FIR High Pass Filter on FPGA

Kaliprasanna Swain[1]
kaleep.swain@gmail.com

Manoj Kumar Sahoo[2]
mksahoo@hotmail.com

Gandhi Institute for Technological Advancement (GITA) /ECE, Bhubaneswar, Odisha, India

***Abstract*— This paper demonstrates the design and implementation of Equiripple linear-phase FIR high pass filter. The filter is modeled using Simulink in Xilinx System Generator. The filter Coefficients are generated with the help of FDA tools, the SysGen tool is used for RTL code generation. Further the model is used as a filter block to interface with ADC/DAC block in VHDL. The design has been prototyped on Spartan-3 DSP protoboard XC3S500fg320 using Integrated Synthesis Environment (ISE) 13.1 tools all in one design suit from Xilinx. Finally the filter is tested by using an audio signal as input and the output is observed in CRO & speaker both.**

***Index Terms*—** Equiripple Filter, FDA Tools, Simulink, Spartan-3, Xilinx System Generator (XSG), FPGA

I. INTRODUCTION

The design of FIR filter using Windows methods leads to good performance filters. However, sometimes there is a need to design an FIR filter that not only performs well but it is optimal. Optimization is the ability to specify a maximum error on each band of interest. This error is expressed as the absolute difference between the ideal or the desired frequency response and the actual or resulting frequency response. One of the techniques to design optimal FIR filters is to minimize a Chebyshev error criterion. The resulting filters are known as Equiripple FIR Filters. It uses the Parks–McClellan and Remez exchange methods for designing a linear-phase (symmetric) equiripple FIR. An Equiripple filter is ideally suited for applications in which a specific tolerance must be met as it has equal ripple in both pass band and stop band.

The frequency response in the passband of FIR filters designed by using the modified Fourier series[1] method has a monotonically decreasing response and a maximum error from the desired ideal response in the passband, at the cutoff frequency ωc. But in this method that "spreads out" the error over the passband in an Equiripple fashion, such that the maximum error is the same at several points and can be made very small. This method minimizes the maximum error in the passband and is called as the *minimax* design or the *Euiripple design*.

A typical filter specification not only includes the specification of passband ωp and stopband ωs frequencies and ideal gains, but also the allowed deviation (or ripple) from the desired transfer function[2]. The transition band is most often assumed to be arbitrary in terms of ripples. A special class of FIR filter that is particularly effective in meeting such specifications is called the equiripple FIR. An Equiripple design protocol minimizes the maximal deviations (ripple error) from the ideal transfer function. The Equiripple algorithm applies to a number of FIR design instances.

II. PROPOSED MODEL

The digital filter can be designed by VHDL/Verilog HDL. In this case reconfiguration method is used so that

Research Paper
First Online on – 30 Dec 2014, Revised on – 30 March 2020

Cite this article – Kaliprasanna Swain and Manoj Kumar Sahoo, "Design and Implementation of Equiripple FIR High Pass Filter on FPGA", *International Journal of Computational and Electronics Aspects in Engineering*, RAME Publishers, vol. 1, issue 1, pp. 1-4, 2014, Revised in 2020.
https://doi.org/10.26706/ijceae.1.1.20141201

the design requires less time for any other application with minimum modification rather than fully structural recoding. This design process first the fixed floating number representation is realized, then converted to an integer based representation which is more suitable for a hardware design process. The FPGA is preferred as a programmable device than any DSP microcontroller due to memory, speed of operation and flexible architecture. The proposed model is given as in figure.1.

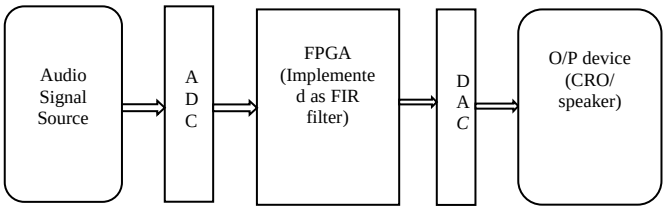

Figure.1 Proposed Model

III. DESIGNING ON SIMULINK

Simulink provides a model based environment for multi-domain simulation. Xilinx System Generator in Simulink environment allows model based system design and analysis with the help of Xilinx blockset and also supports automatic code generation, system-level design simulation, and continuous test and verification of FPGA based embedded systems. Simulink is integrated with MATLAB to provide a graphical editor, customizable block libraries, and solvers for modeling and simulating dynamic systems. It also enables us to incorporate MATLAB algorithms into models and export simulation results to MATLAB for further analysis. The entire design as realized in Simulink is shown in Figure. 2. The constituent blocks are discussed in the preceding text. The various steps [3] [4] used for implementation methodology according to flow graph are as follows.

A. Design the Filter with FDA tool

The Filter Design and Analysis Tool (FDA Tool) is a powerful graphical user interface (GUI) in the Signal Processing Toolbox™ for designing and analyzing digital filters of required specification. The different response types such as highpass, lowpass, bandstop, differentiator, integrator etc. and different design methods such as IIR, FIR are available to implement the filter. These windows can be customized by providing order of the filter, cutoff, sampling, the pass-band and stop-band frequencies and magnitude specifications. By quickly setting filter performance specifications, by importing filters from MATLAB® workspace or by adding, moving or deleting poles and zeros, digital FIR and IIR filter can be designed using FDA tool. It can also be used for analyzing filters, such as magnitude and phase response plots and pole-zero plots. Design specifications for low pass filter using FDA tool is given in table 1.

TABLE 1

FDA TOOL SPECIFICATION

Response type	Highpass
Order of filter	120
Passband Frequency, fp	10kHz
Stopband frequency, fs	15kHz
Sample frequency, fsamp	250kHz
Density factor	16

B. Creation of Simulink model

The Xilinx System Generator for DSP [5-7] is a system level modeling and design tool that facilitates for Xilinx FPGA design and has the ability to work at a higher level of abstraction. It enables the use of the Mathworks graphical model based Simulink design environment for FPGA design. The System Generator integrates itself with Simulink and FPGA designs are captured by using the Xilinx specific Blockset. Thus, designing a hardware model in Simulink is as simple as designing any other Simulink model with the only difference being the use of Xilinx Blockset instead of those found in Simulink. The System Generator uses the Xilinx ISE software and Xilinx IP cores generators to convert a designed model into the equivalent HDL code. The remaining FPGA implementation steps, including synthesis, place and route, etc. are automatically performed to generate a bit file that is downloaded onto the FPGA.

Fig 3 shows the top module of both the filter using Xilinx System Generator.

- The gateway in blocks: The inputs into the Xilinx portion of the Simulink design considering as fixed point data in binary, Boolean and discrete type for a digital design.
- The gateway out blocks: The outputs from the Xilinx portion of the Simulink design, it manages all digital data type with proper buffering.

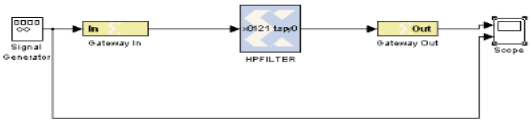

Figure.2 Top module of Highpass filter system using XSG

C. Generation of Code using SysGen

Xilinx System Generator (XSG) as a code generator provides an equivalent synthesizable HDL code, which is used as a component or sub-system in Xilinx ISE. The generation of the code depends on the block and supported IP use in the design for an FPGA family. According to this, the structure of a design is built up. Fig.3 shows the code generated by XSG.

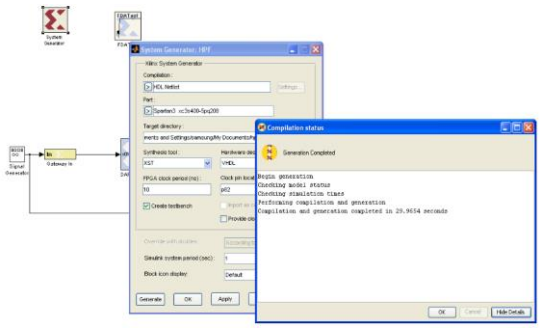

Figure.3 Netlist generation for High pass filter

III. SIMULATION AND RESULT

The logic program is simulated and debugged and necessary correction is made to meet the output. Further RTL schematic is generated and resource utilization is checked as shown in figure. 4 and 5.

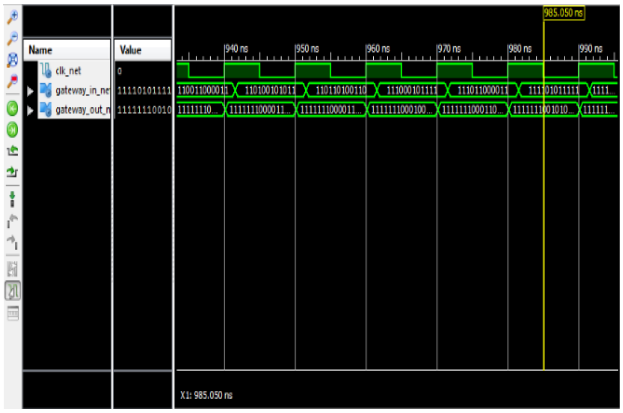

Figure.4 Highpass filter Simulation

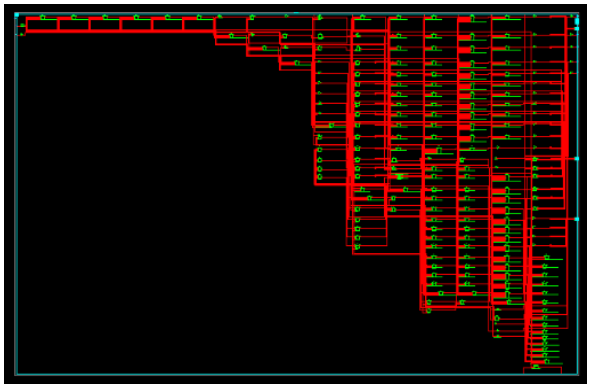

Figure.5 Highpass RTL Design

An automatic place and route program was run to place the logic block in appropriate places in FPGA and then the interconnection between logic blocks were routed. Figure. 6 and 7 has shown the device summary and the area utilized for highpass filter.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	6,049	7,168	84%	
Number of 4 input LUTs	3,939	7,168	54%	
Number of occupied Slices	3,206	3,584	89%	
Number of Slices containing only related logic	3,206	3,206	100%	
Number of Slices containing unrelated logic	0	3,206	0%	
Total Number of 4 input LUTs	4,045	7,168	56%	
Number used as logic	3,871			
Number used as a route-thru	106			
Number used as Shift registers	68			
Number of bonded IOBs	35	141	24%	
IOB Flip Flops	23			
Number of BUFGMUXs	2	8	25%	
Average Fanout of Non-Clock Nets	1.92			

Performance Summary			
Final Timing Score:	0 (Setup: 0, Hold: 0, Component Switching Limit: 0)	**Pinout Data:**	Pinout Report
Routing Results:	All Signals Completely Routed	**Clock Data:**	Clock Report
Timing Constraints:	All Constraints Met		

Figure.6 Highpass device summery

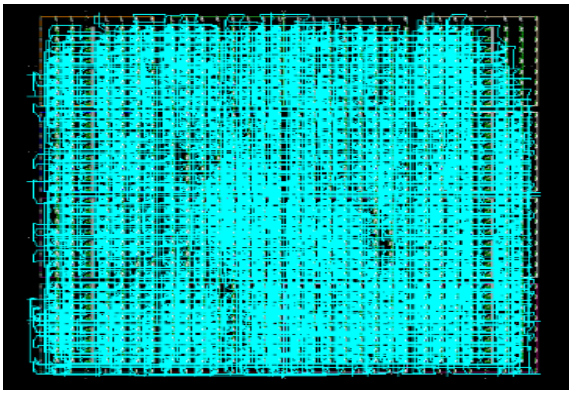

Figure 7. Area utility figure for high pass

After the implementation was carried out on Spartan-3 DSP protoboard XC3S500fg320 using Integrated Synthesis Environment (ISE) 13.1 tools, an audio signal was given as Input and the Output was observed on a CRO and Speaker separately as shown in experimental setup (figure. 8).

Figure. 9 and 10 have shown the result when the audio signal is passed before and after the HPF.

Figure 8. Area utility figure for high pass

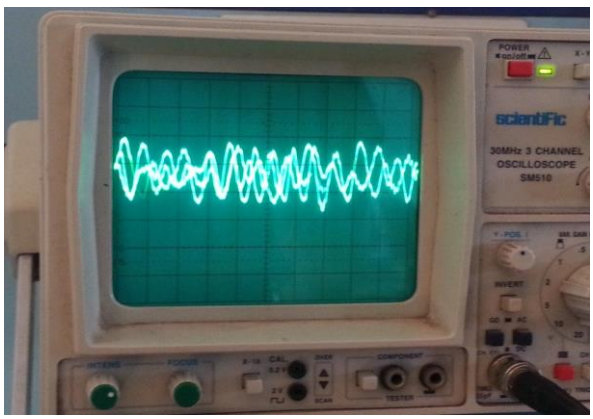

Figure 9. Before highpass filtering

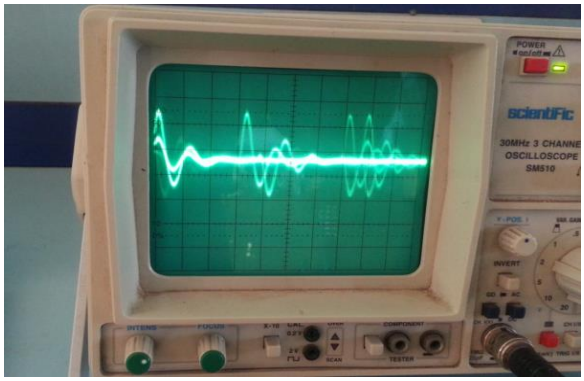

Figure 10. After highpass filtering

IV. Conclusion

Design and implementation of equiripple linear-phase FIR low pass filter is thoroughly discussed in this paper. Simulink in Xilinx system generator is used for the same. It is revealed that the efficiency of filtering can be increased by using a higher sampling rate (ADC/DAC) and the bulkiness of the filter can also be decreased by using sub-band filtering method. The above design can be modified or redesigned as an all pass digital filter with a single output mode or multi output mode.

References

[1] B. A. Shenoi, Introduction to digital signal processing and filter design, (JOHN WILEY & SONS, INC., 2006) pp.280-285.

[2] UweMeyer - Baese, Digital signal processing with field programmable gate arrays, 3rd ed. (Tsinghua University Press, Beijing, 2007), pp. 175 - 177.

[3] B. Mamatha1, V.V.S.V.S. Ramachandram, "Design and implementation of 120 order fir filter based on FPGA", International Journal of Engineering Sciences & Emerging Technologies, August 2012,Volume 3, Issue 1, pp. 90-97.

[4] Harish V. Dixit , Dr. VikasGupta,"IIR filters using Xilinx System Generator for FPGA Implementation", IJERA,Vol. 2, Issue 5, September- October 2012, pp.303-30.

[5] ISE 13.2, Quick Start Tutorial, Xilinx.

[6] Xilinx System Generator for DSP User Guide, r10.1.1, April 2008.

[7] Nexys3™ Board Reference Manual Revision: December 28, 2011, www.digilentinc.com