IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 1, Issue 4, Aug-Sept, 2013
ISSN: 2320 - 8791
www.ijreat.org

Field Programmable Gate Array Implementation of Digital Filter of Highest-Possible Order and its Testing using Advanced Microcontroller

Dr. Pawan K. Gaikwad
Head and Assistant Professor in Electronics
Willingdon College, Sangli (M.S.) INDIA

***Abstract*—**The present research paper depicts a technique estimating the Field Programmable Gate Array (FPGA) resources utilization for implementation of digital filters with different orders. For a low pass Butterworth filter, excluding its order, rest of the design parameters were kept constant. Filter designing was carried out using the numerical computing environment software, MATLAB. Its outstanding facility generating Hardware Description Languages (HDLs) for a digital filter object was deployed. Different Very High Speed Integrated Circuit HDL (VHDL) source codes were generated and implemented in Xilinx FPGA device Spartan-3E by increasing filter-order, until the device utilization exceeds the available resources. In this way, a maximum 18^{th} order filter was implementable in FPGA. The digital filter response was displayed on an Oscilloscope by means of a microcontroller, Advance RISC Machine (ARM) device LPC2148. The analogue to digital and digital to analogue conversion over digital filter data was performed in a single ARM chip.

***Keywords*—**ARM, Digital filter order, FPGA resources, Sharp cut-off.

I. INTRODUCTION

DIGITAL Signal Processing (DSP) systems are becoming more sophisticated and demanding greater computational performance with optimized resource utilization. There are several DSP processor chips available in the market; on the contrary, most of them are bound to their relevant factors like execution time, memory space utilization, and power consumption. Integrated Circuit (IC) manufacturers are marching on the technological track that continues to develop enhanced DSP processors on the same silicon substrate, involving system components like microcontrollers and smart peripherals.

Digital filters are the main processing unit of a DSP system. The region between the pass- and stop-band is referred to as the transition band or transition width. This width (in Hz) depends on how sharply the filter response drops from the pass band to the stop band. Related to this is the roll-off rate, which, for low-pass filters is the rate at which the signal gain decreases when the signal is above the cut-off frequency. The narrower the transition band, the steeper the roll-off [1]. The digital version of low pass Butterworth filter is having maximally flat frequency response in the vicinity of the zero frequency and has a transition band, cantered on chosen cut-off frequency that can be narrowed by increasing the filter order [2]. However, the trade-off between transition-bandwidth and filter-order is critical, since it dictates both memory and processing resources needed in real-time DSP [3].

As systems become more complicated and processor choices grow, designer needs good estimates of a processor's DSP performance. The methodology discussed in [4] is one of the starting places for calculating such estimates. On the other hand, Field Programmable Gate Array (FPGA) design technology has becoming the preferred platform for evaluating and implementing signal processing algorithms. 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 application specific integrated circuit (ASIC) for moderate volume applications, and more flexibility than the alternate approaches. Since many current FPGA architectures are in-system programmable, the configuration of the device may be changed to implement different functionality if required [5].

The technique presented in this research work may help DSP processor engineers, who wish to design digital filters. It is not only to implement filters on FPGA, but also help preserve the device resources for the architectures to be augmented, if desired. A low pass Butterworth filter was chosen for this research work. Out of various filter design parameters, the filter order was made changing to find the percentage of resources utilization from a selected FPGA device. Filter designing was carried out using the MATLAB software. The frequency response analysis for various filters; having different orders, was carried out using the Filter Visualization Tool, *fvtool*, included in the MATLAB itself. The MATLAB has a provision of generating Hardware Description Languages (HDLs) like Very High Speed Integrated Circuit HDL (VHDL), and Verilog for a digital filter object. VHDL files were generated by varying only the filter order in MATLAB file [6] used to design a low pass Butterworth filter. A top level entity was developed to get

instantiated such filters' VHDL codes, one by one. Each of the times, the design was *synthesized* and *implementation* process was performed by an Electronic Design Automation (EDA) tool, Xilinx Integrated Software Environment (ISE) Design Flow. The Device Utilization Summary was noted for each filter-order, varying in the division of two, and in ascending order. It was observed that, if any of the device utilization factors of a selected FPGA device exceeds the figure 100%, the Xilinx ISE tool synthesizes the design, but does not support the *implementation*.

The final testing of an FPGA-implemented filter was carried out providing different frequency signals and monitoring the filter response on a Digital Storage Oscilloscope (DSO). The sinusoidal signal of variable frequency was converted into its digital form using Analogue to Digital Convertor (ADC), available on Advanced RISC Machine (ARM) microcontroller chip. Such digital output was connected as an input to the filter implemented in the FPGA device. The filter response, emerging from FPGA output was further converted into its analogue form using a Digital to Analogue Convertor (DAC) available on the chip used for ADC process. The sinusoidal input and output waveforms of the filter were observed on DSO; displaying signal parameters in terms of amplitude and frequency.

II. Filter Design and FPGA Implementation using Xilinx ISE Design Flow

The Fig.1 shows essential course of actions followed for a target filter design and fine-tune the suitable filter-order with the logic resources existing within a selected FPGA.

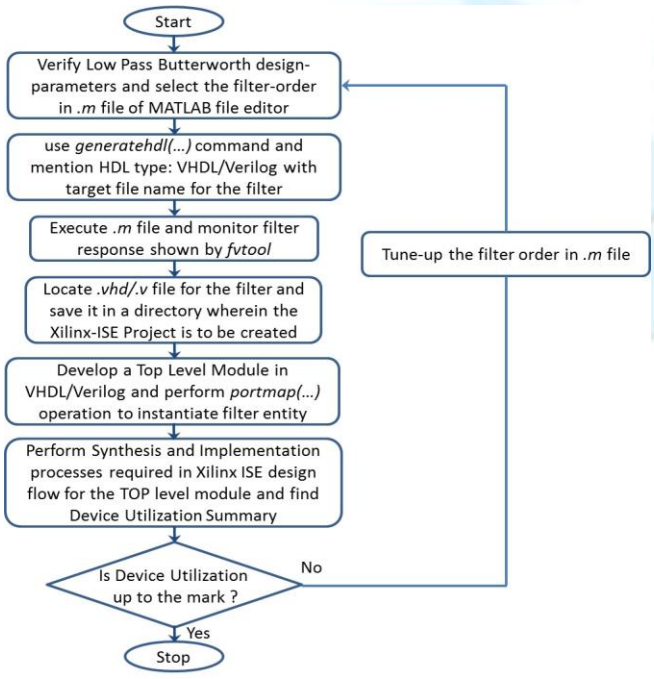

Fig. 1 A flow diagram to design the digital filter with an order; suitable in the available FPGA logic resources

A low pass Butterworth filter was selected for its implementation in Xilinx FPGA. Out of various digital filter-design parameters, like sampling frequency, Nyquist rate and cut-off frequency were modified in the MATLAB file [6], and a filter object was generated. As directed in MATLAB program file [6], after following to the process of creating Quantized filter and re-quantization of the same, the filter responses were studied using Filter Visualization tool, *fvtool*. A noteworthy facility provided by MATLAB tool to generate HDL for a design object was used, and VHDL source code was generated for the filter object using *generatehdl* command.

As shown in the Fig. 1, the filter VHDL file was saved in a directory on the computer hard-drive. The Xilinx ISE Project was created for a top level VHDL entity in a directory, wherein, the filter VHDL code was stored. (Alternatively, it is also possible to add the filters' source file located anywhere on the computer hard-drive into the Xilinx ISE Project directory.) The filters' HDL-component (VHDL code) was instantiated into the top level module. The process of *Synthesis* and *Implementation* of such a module was executed, as necessary in the Xilinx ISE Design Flow. The Device Utilization Summary, including various logic utilization factors were studied by exploring the Synthesis report of the Xilinx ISE tool. Different *Synthesis reports* were generated for digital filters; as their order was modified in the MATLAB File Editor. While changing the filter-order, rest of the design parameters as given in [6], were kept untouched, and the analysis was focused on changes in the FPGA device resources utilization. On the other hand, there are other filter-design factors too, that could affect the FPGA device utilization. The flow diagram shown in Fig. 1 depicts that, when a DSP processor design engineer come across the proper tuning between the FPGA resources utilization and possible maximum filter-order, he has to stop the process of changing the filter-order and generating VHDL files.

A. Frequency Response Analysis for the Filters with Different Orders using MATLAB Software

The Fig. 2 depicts nine different low pass Butterworth filter-responses and reveals that, if there is an increase in the filter-order, its pass-band response remains sustained more, than the lower order filter; thereby reducing the transition bandwidth and sharpening the roll-off. However, in this research work, a bottleneck originated while implementing a filter of 20^{th} order into the Spartan-3E FPGA; due the existing limited resources of the selected Xilinx device. Therefore, the low pass Butterworth filter responses of this particular research work are shown up to the 18^{th} order filters in Fig.2. A transition curve, from each of the filters' response-view has been focused by windowed horizontal and vertical axes, instead of the entire response. The frequency values along the horizontal axis are plotted in the range of 200Hz to 750Hz; except in the case of second order filter (beginning at 100Hz), because, its pass

band response starts decrease earlier than the other (high order) filters. The filter output magnitude values are plotted on the vertical axis with -4 decibels (dB) through 0dB. The responses with zero decibels indicate that, there is no attenuation to pass the digital filters' input signal, and produces at the output.

To produce such nine filter responses using *fvtool*, a single MATLAB file was developed using lines of MATLAB file [6], with modifications of filter order 'n' like n=2, 4, 6,…18.

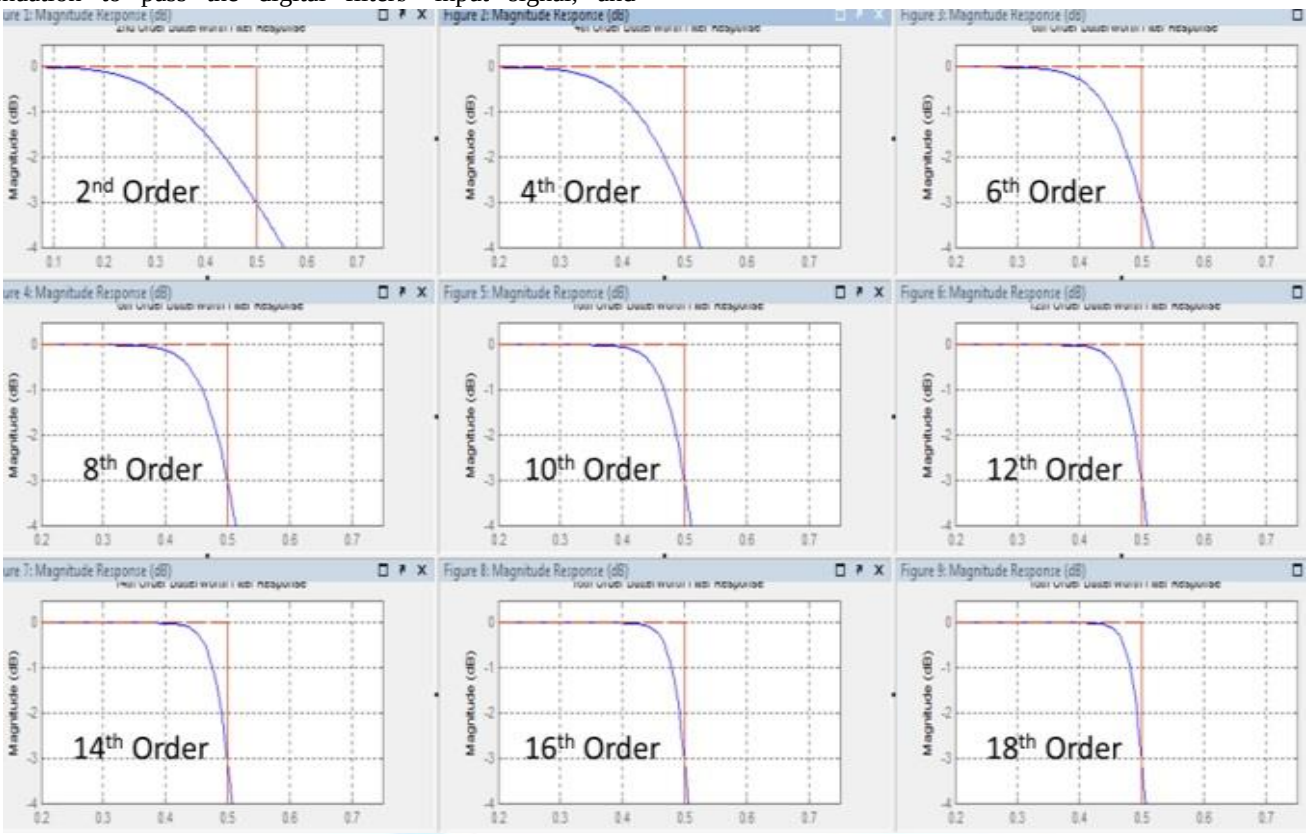

Fig. 2 The MATLAB Filter Visualization Tool, *fvtool*, shows frequency responses for a Low Pass Butterworth filter of 2nd order to 18th order (in division of 2) and illustrates that, an increase in the filter-order, reduces the transition bandwidth

The Fig. 3 shows a focused view of the 18th order filter response curve. A diversion of the *actual filter response* line from a straight-nature to the curve indicates an earlier starting of the attenuation than that of the *ideal filter response* line. The magnitude and frequency values, marked at point 'A', on the *actual filter response* curve line are -0.00310871dB and 0.4089355KHz respectively. It indicates there is almost no attenuation below the cut-off frequency. On the contrary, as shown at point 'C' in Fig.3, the intercept of *actual filter response* line with the *ideal filter response* line indicates the location of cut-off frequency 0.5004883 KHz, at which the attenuation is of the order of -3.087325dB; onward to which the stop band begins. The research work does not emphasis only on how long the pass band can sustain before the cut-off frequency, but also there is a quantitative analysis of the extent of FPGA resources, required to implement the best possible high-order filter aiding sharp cut-off.

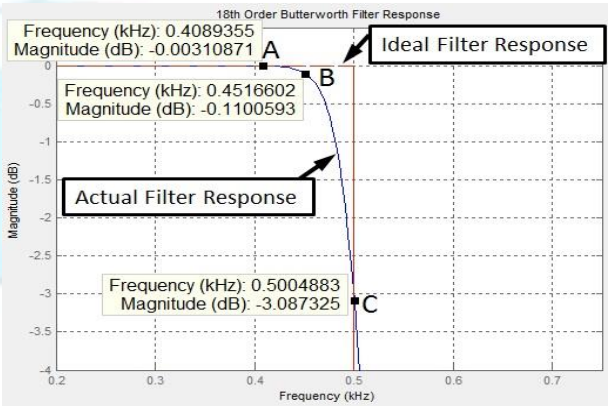

Fig. 3 Detailed response curve analyses of 18th order low pass Butterworth filter transition bandwidth

III. Resource Utilization in FPGA for Implementation of Low Pass Butterworth Filter of Highest-Possible Order

The imperative design parameters like Sampling frequency, Nyquist frequency, Cut-off frequency, and filter-order were provided into the MATLAB file [6] editor, to create an object

for the low pass Butterworth filter. The filter object was set to a fixed-point mode for quantization purpose and it was recycled to create a VHDL entity. The numbers for input and output lines for this entity were set to 10 bits, and the same figure was customized for the filter coefficient word length. The length of *adders* and *states* was set to 20 bits. After creating such a filter object, its performance was monitored on Frequency Visualization tool, *fvtool*. The filter object was re-quantized to get the better response by changing coefficient word length from 10 to 16. The essential scaling was performed for hardware development of the filter; because, a key step for hardware realization of the filter design is to check whether the scale values are reasonable and adjust the scale value if needed [6].

An entity named as *lpbutter18thOrder* was generated, and its declaration is shown in following lines of VHDL code.

```
ENTITY lpbutter18thOrder IS
 PORT( clk : IN    std_logic;
 clk_enable : IN    std_logic;
 reset      : IN    std_logic;
 filter_in  : IN    std_logic_vector(9 DOWNTO 0);
 filter_out  : OUT   std_logic_vector(9 DOWNTO 0)  );
END lpbutter18thOrder;
```

The signal names of the entity *lpbutterOrderTOP* are in capital letters and, *lpbutter18thOrder* entity signal names are in small letters. The present research work targets the Xilinx FPGA device XC3S500E, which is the family member of Spartan-3E and has package type of FG320. The FPGA chip is embedded on a development board Nexys2, developed by Digilent Inc. It has a provision of on-board clock of 50MHz. This clock is given to *CLK* input of the top level module *lpbutterOrderTOP*. Inside this entity, a clock divider module is instantiated as shown in Fig. 4, it divides 50MHz clock and generates a 50KHz frequency signal. The 50KHz frequency signal is routed towards the clock input, *clk* of the filter entity *lpbutter18thOrder* by using *port map* syntax of VHDL. In other words, the sampling process over a 10-bit digital input, *filter_in*, was performed at the rate of 50KHz. Thus, in addition to the filter, another entity, that performs frequency division by 100, was instantiated in the top level module *lpbutterOrderTOP*. (The frequency divider module requires not more than 1% of the total resources available in the FPGA device.) The top entity signal *CLK_ENABLE* was asserted *high* and the reset signal *RESET* was made disabled during the sampling process. In this way, the digital filter output (connected to top module) was obtained at the 10-bit output vector line *FILTER_OUT*.

The Fig. 4 shows a top level entity *lpbutterOrderTOP* and depicts an internal view of Register Transfer Level (RTL) *synthesis* result, obtained using an EDA tool from Xilinx ISE 14.1 version. This involves two VHDL modules; first one is *clkDivMain* that originates sampling frequency 50KHz from a 50MHz clock to run the Butterworth filter entity.

TABLE I
RESOURCES AVAILABLE IN XILINX FPGA SPARTAN-3E, XC3S500E-4FG320

Type of Logic Resources	Available
Number of Slice Flip Flops	9,312
Number of 4 input LUTs	9,312
Number of occupied Slices	4,656
Number of Slices containing only related logic	220
Number of Slices containing unrelated logic	220
Total Number of 4 input LUTs	9,312
Number of bonded IOBs	232
Number of BUFGMUXs	24
Number of MULT18X18SIOs	20

The similar methodology was followed to perform FPGA-implementation of the other low pass Butterworth filters; by changing their order in the same MATLAB file.

TABLE II
DEVICE UTILIZATION SUMMARY REGARDLESS OF FILTER ORDER

Type of Logic Resources	Utilization
Number of Slices containing only related logic	100 %
Number of Slices containing unrelated logic	0 %
Number of bonded IOBs	9 %
Number of BUFGMUXs	8 %

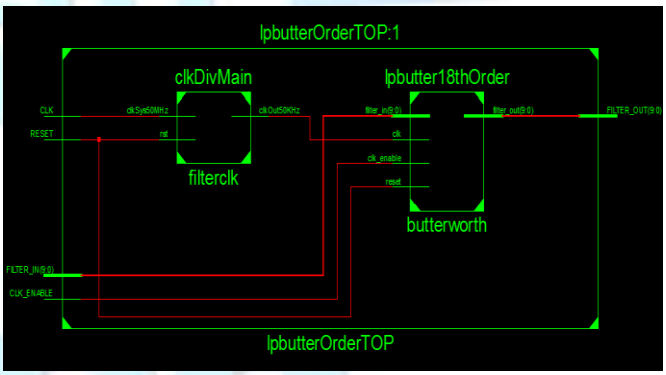

Fig. 4 Internal view of Synthesis results at Register Transfer Level (RTL) covering two entities *clkDivMain* and *lpbutter18thOrder* with their inter-connections within the top level module *lpbutterOrderTOP*

The Table I illustrates different resources available within the Xilinx FPGA Spartan-3E device: xc3s500e-4fg320. These resources were utilized to implement VHDL entity of the filter and the clock divider module together.

After the process of *synthesis* and *implementation* of the top level module *lpbutterOrderTOP*, the Xilinx Synthesis report shows that, there are some FPGA resources, having no variations in their percentage of logic utilization; even there is a change in the filter order. Such invariant values of FPGA resources are shown in Table II.

On the other hand, as given in Table III, some logic resources utilization factors were found rising by increase in

TABLE III
XILINX FPGA SPARTAN-3E DEVICE LOGIC RESOURCES UTILIZATION VARIATION DUE TO CHANGES IN THE FILTER ORDER

Type of Logic Resources Utilized	Filter order n=2	Filter order n=4	Filter order n=6	Filter order n=8	Filter order n=10	Filter order n=12	Filter order n=14	Filter order n=16	Filter order n=18	Filter order n=20
Number of Slice Flip Flops	1 %	1 %	2 %	2 %	2 %	2 %	3 %	3 %	3 %	4 %
Number of 4 input LUTs	2 %	5 %	17 %	29 %	42 %	50 %	67 %	70 %	80 %	100 %
Number of occupied Slices	4 %	9 %	23 %	38 %	52 %	63 %	82 %	88 %	99 %	117 %
Total Number of 4 input LUTs	3 %	7 %	21 %	35 %	49 %	60 %	79 %	84 %	95 %	116 %
Number of MULT18X18SIOs	35 %	85 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %

the filter order; designated here as 'n', in division of two i.e. 2,4,6 and so on, up to 20. It also depicts that, when 20th order filter (n=20) was chosen for FPGA-implementation, two important factors: *Number of occupied Slices* and *Total Number of 4 input Look-Up Tables (LUTs)* surpasses the existing FPGA logic resources and strive for *117%* and *116%* logic requirement, respectively. Xilinx ISE tool produces two error messages in such a case, as

ERROR: Pack: 2310 - Too many comps of type "SLICEL" found to fit this device

ERROR: Pack: 18 - The design is too large for the given device and package.

Therefore, moving a step back from 20th order, the 18th order filter was chosen for implementation in the FPGA. The final testing for the 18th order (highest possible for FPGA Spartan-3E device) low pass Butterworth filter was performed to realize the system. As per the Xilinx ISE Design Flow, the process of *synthesis* and *implementation* for the top level entity was performed. A user constrain file (.ucf) for the same module was also developed for *place* and *route* processes. Due to this it perform interconnections between Inputs and Output (I/O) signals of VHDL entity *lpbutterOrderTOP* and associated FPGA pins. A program file (named as *lpbutterOrderTOP.bit)* was generated to configure into the on-board Platform Flash ROM via USB cable. Adept software, provided by Digilent Inc., was deployed to download such a *bit stream* file into the Flash ROM using the process as given in [7].

IV. FPGA IMPLEMENTED FILTER RESPONSE ANALYSIS ON OSCILLOSCOPE USING MICROCONTROLLER: ADVANCED RISC MACHINE

To provide a 10-bit input signal to the FPGA based digital filter, as well as to monitor its frequency response at the 10-bits output, the ADC and DAC from a microcontroller ARM were deployed. Fig. 5 shows a logic diagram, depicting interface of the microcontroller ARM7-LPC2148, and the Xilinx device FPGA Spartan-3E. For software development, two Embedded C source codes, as given in [8] were modified and combined in a way to drive ADC and DAC in a single ARM chip. The process of compilation and debug of such a source code was performed by an EDA tool, Kiel (Micro version-4).

TABLE IV
INTERFACING LINES OF MICROCONTROLLER ARM7 IC-LPC2148 AND XILINX FPGA SPARTAN-3E BOARD: NEXYS2

ARM7 LPC2148 I/O Port lines	Digital Filter Top level Entity Signal Names	FPGA I/O Pins	FPGA board Pmod Connector Pins[a]
P0.0	FILTER_OUT[0]	L15	JA1
P0.1	FILTER_OUT[1]	K12	JA2
P0.2	FILTER_OUT[2]	L17	JA3
P0.3	FILTER_OUT[3]	M15	JA4
P0.4	FILTER_OUT[4]	K13	JA7
P0.5	FILTER_OUT[5]	L16	JA8
P0.6	FILTER_OUT[6]	M14	JA9
P0.7	FILTER_OUT[7]	M16	JA10
P0.8	FILTER_OUT[8]	M13	JB1
P0.9	FILTER_OUT[9]	R18	JB2
P1.16	FILTER_IN[0]	T18	JB9
P1.17	FILTER_IN[1]	U18	JB10
P1.18	FILTER_IN[2]	G15	JC1
P1.19	FILTER_IN[3]	J16	JC2
P1.20	FILTER_IN[4]	G13	JC3
P1.21	FILTER_IN[5]	H16	JC4
P1.22	FILTER_IN[6]	H15	JC7
P1.23	FILTER_IN[7]	F14	JC8
P1.24	FILTER_IN[8]	G16	JC9
P1.25	FILTER_IN[9]	J12	JC10

[a]Pmod Pins JA5, JA11, JB5, JB11, JC5, JC11 are given to the Ground point and JA6, JA12, JB6, JB12, JC6, JC12 to DC power line (3.3V) on the Board

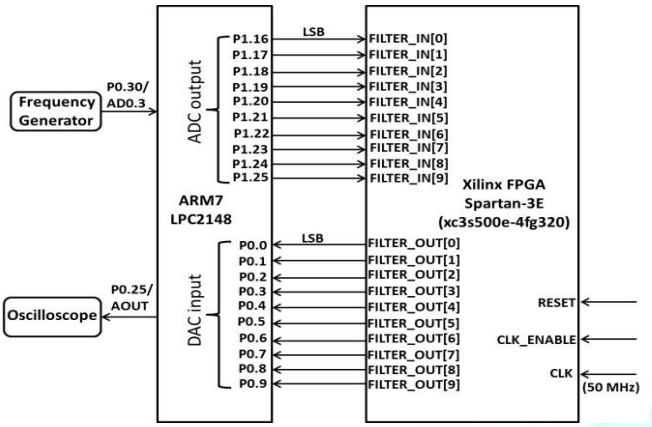

Fig. 5 Logic diagram showing interfacing of Advanced RISC Machine and Xilinx FPGA Spartan-3E

As shown in Fig. 5, a sinusoidal signal generator is connected to the ARM device. The ARM-ADC converts the analogue signal (connected at its P0.30/AD0.3 pin) into the 10-bit digital output and produces at the pins P1.16 through P1.25. These lines are input to FPGA-filter signals FILTER_IN[0] through FILTER_IN[9]; associated to the top level entity *lpbutterOrderTOP*. In response to such input, FPGA generate filters' output at FILTER_OUT[0] through FILTER_OUT[9] signals. These are further connected to the input lines of ARM-DAC, that is, P0.0 through P0.9 to convert back into its analogue form. Thus, sinusoidal amplitudes were regenerated at the P0.25/AOUT pin of the microcontroller.

On the other hand, a clock signal CLK (of top entity) was connected to the FPGAs' dedicated clock pin, designated as B8. The *RESET* and *CLK_ENABLE* signals were also connected to FPGA I/O pins *B18* and *G18*, respectively, because these pins are further linked to on-board switches. The FPGA Nexys2 board has four, 12 pin connectors called as *Pmods*, and labeled as JA, JB, JC, and JD, for external hardware interface. Pin-out for each *Pmod* connector is provided in [9]. The Table IV illustrates inter-connection details of the microcontroller I/O lines, and the FPGA I/O pins (as shown in Fig. 5); which are associated with Nexys2 on-board *Pmod* connectors.

The Fig. 6 illustrates three snapshots combined together, which were captured at the time of monitoring sinusoidal signals for three different input frequencies. The signal generator waveforms are shown at the channel 1(CH1), and simultaneously, the output of digital filter (converted into analogue form by ARM-DAC) is also shown at channel 2 (CH2) of the oscilloscope. As previously shown in Fig. 3, there are three marking points, 'A', 'B', and 'C', which makes good understanding of filter response; especially, when the response curve resumes entering in the phase of attenuation and approach towards the cut-off location. Therefore, frequency values of the order of 408Hz, 449.9Hz, and 500Hz (pertaining 'A', 'B', and 'C' points in Fig .3) were chosen to apply as input to the digital filter. Three waveforms of Fig. 6 reveal that, output amplitude for input frequencies 408Hz and 449.9Hz are 80mV and 74mV, respectively. The amplitude drops down to 52mV at the cut-off frequency, 500Hz.

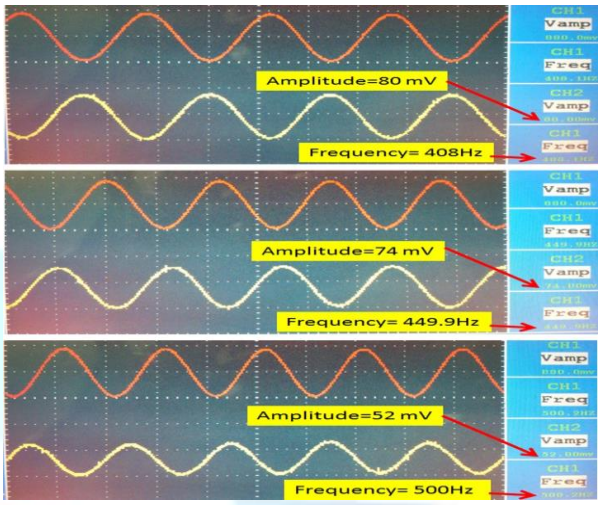

Fig. 6 Digital Storage Oscilloscope (DSO) displaying sine to sine signal of the maximum amplitude for pass band frequencies, and decreasing while tending towards the cut-off

The Fig. 7 shows the entire FPGA and ARM interfacing setup, connected with a signal generator and DSO. A minimum system of the microcontroller ARM is also shown. It has analogue input line, which is connected at ADC pin P0.30/AD0.3 of the device. To monitor amplitude of analogue signal, which was generated by ARM-DAC, a DSO was connected to analogue output line of the microcontroller. In this way, input as well as output signals of a digital filter, implemented in the FPGA, were monitored on a digital storage oscilloscope in the form of sinusoidal waves by means of the ARM-ADC and DAC both functioning in a single chip.

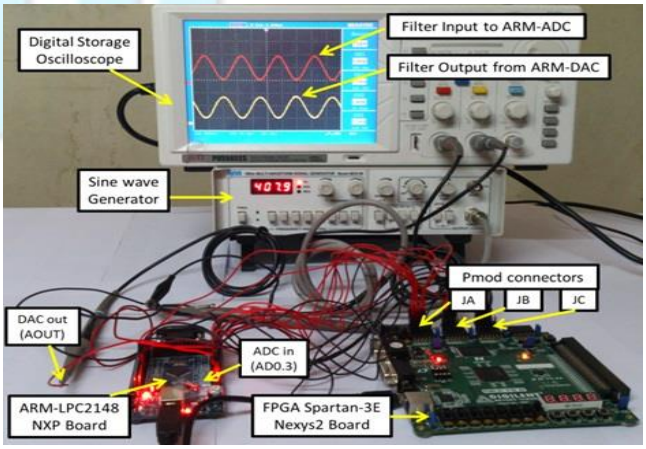

Fig. 7 A photograph taken at the time of monitoring FPGA-implemented filter response using microcontroller Advanced RISC Machine (ARM)

ACKNOWLEDGMENT

The present paper is a combination of two technologies, like FPGA based system, and another is ARM7 platform; used for Digital Filter testing. The former focuses estimating the FPGA logic resources management for a particular filter order, and latter is to test the digital filter and visualize its performance on a digital storage oscilloscope. The digital filters are somewhat difficult to imagine in theoretical way; but are enjoyable while performing practical on them.

REFERENCES

[1] Tech. Note, "Signal Filtering-09A", ADInstruments Pty Ltd., (2009), pp.1-7.
[2] D.S.G. POLLOCK, "ECONOMETRIC METHODS OF SIGNAL EXTRACTION", University of London and GREQAM: Groupement de Recherche en Economie Quantitative d'Aix-Marseille, p.5.
[3] Woon S. Gan, Woon-Seng Gan and Sen M. Kuo, "Transition from Simulink to MATLAB in Real-Time Digital Signal Processing Education", School of Electrical and Electronic Engineering Nanyang Technological University, Singapore 639798, pp.6-7.
[4] "Evaluating DSP Processor Performance", Berkeley Design Technology, Inc., (1997-2000).
[5] Rakhi Thakur and Kavita Khare, "High Speed FPGA Implementation of FIR Filter for DSP Applications", International Journal of Modeling and Optimization, Vol. 3, No. 1, February 2013, pp. 92-94.
[6] MathWorks Document, "HDL Butterworth Filter", Documentation Centre, Viewed 16 March 2013, from http://www.mathworks.in/help/hdlfilter/examples/hdl-butterworth-filter.html.
[7] "Adept Software Basic Tutorial", Digilent, Inc., 215 E Main Suite D | Pullman, WA 99163, (509) 334 6306, Doc: 594-006, Revision: February 26, 2010.
[8] jarm7-lpc2148, ARM7 LPC2148 Based project Development for Beginners viewed 5 August 2012, from https://code.google.com/p/jarm7-lpc2148/downloads/detail?name=JARM7-LPC2148-Source.rar&can=2&q=
[9] Reference Manual, "Digilent Nexys2 Board", Digilent, Inc., 215 E Main Suite D | Pullman, WA 99163, (509) 334 6306, Doc: 502-134, Revision: July 11, 2011.

P.K. Gaikwad born in Ashta (K), Osmanabad District, Maharashtra, INDIA, on August 29th, 1976, and is M.Sc. in Electronic Science, from Department of Electronic Science, University of Pune, Pune (1999), Ph.D. in Electronics, from Shivaji University, Kolhapur (2010), Diploma in VLSI Design, from Silicon Magic Tech. Pune (2000), Diploma in Cyber Laws, from Asian School of Cyber Laws, Pune, Maharashtra, INDIA.

He is **Head of the Department, and Assistant Professor in Electronics** at Willingdon College, Sangli, Maharashtra, INDIA, and in teaching profession since last 13 years. He is working as a **Research Guide in Electronics** for Shivaji University, Kolhapur. At present three Ph.D. students are working under his guidance. He has published six research papers at International level Journals and Five papers at National level Journals. He is an instrumental author of a book: "Harnessing VLSI System Design with EDA Tools", Dordrecht Heidelberg London New York, Springer, 2011. His working platform and area of research interest is the development of Embedded and VLSI based systems; especially which are applicable in Biomedical instrumentation, such as Electrocardiography (ECG), Electroencephalography (EEG).