ISSN:2321-1156
International Journal of Innovative Research in Technology & Science(IJIRTS)

IMPLIMENTATION OF HP FIR FILTER ON FPGA USING DA

Kavya Jyothi.B, 4th sem, M.Tech,(VLSI and Embedded systems), SSIT, Tumkur 1;
Dr. K.B.Shivakumar, *Professor*, Dept of Telecommunication Engineering, SSIT, Tumkur, Karnataka 2;
Dr.M.Z.Kurian, Dean & HOD, Dept of E&C, SSIT, Tumkur, Karnataka 3

Abstract

DA (Distributed Arithmetic) is the key for FIR filter implementation on FPGA. Compared with the traditional multiply-add structure DA has the characteristics of pipelined data processing and efficient operation. In this paper design and implementation of HP FIR filter of order 17 on FPGA using DA is proposed. Developing verilog code for the proposed design and simulating it on simulators is carried out. Simulation experiments and time series analysis are carried out in modelsim and ISIM. The design is verified by downloading to Chipscope Pro on FPGA, Virtex-5. The design of FIR on FPGA has practical applications in digital signal processing.

1. Introduction

The purpose of filter is to enhance the wanted signal relative to unwanted signals, interference or noise. Filters are broadly classified as analog and digital filters. Analog filters are designed using components such as resistors, capacitors, opamps etc. In case of digital filters these components are replaced by filter coefficients. Digital filters are classified into 2 categories:
(a)IIR(Infinite Impulse Response) filter.
(b)FIR(Finite Impulse Response) filter.
IIR filters have impulse response of infinite duration. They have feedback. Hence they can replace analog filters and they are not stable. They do not guarantee linear phase response.

FIR filters have impulse response of finite duration. They do not have any feedback, hence they are stable. Since output is a delayed version of input without any distortion they guarantee linear phase response.

The proposed design uses DA to reduce the number of multipliers required at the expense of additional adders. Exchanging multipliers with adders is advantageous because adders weigh less than multipliers in terms of silicon area. In addition the overhead from the increase in adders in preprocessing and post processing blocks stay fixed, not increasing along with the length of the FIR filter, whereas the number of reduced multipliers increases along with the length of the FIR filter.

2. FIR Filter and Distributed Arithmetic

2.1 FIR filter

Is a filter whose impulse response (or response to any finite length input) is of finite duration, because it settles to zero in finite time. The output *y* of a linear time invariant system is determined by convolving its input signal *x* with its impulse response *h*.

$$y(n) = \sum_{i=0}^{N-1} h(i)x(n-i)$$

Here HP FIR filter of order 17 is designed using Kaiser window. The above equation corresponds to realization of FIR filter using fully parallel architecture. It requires N number of adders i.e., 17. (N+1) number of multipliers and filter coefficients i.e., 18. The number of memory locations required will be 2^{18}.

2.2 Distributed Arithmetic

Distributed arithmetic is an efficient procedure for computing inner products between a fixed and a variable data vector.
Consider the sum-of-products

$$Y = \sum_{i=1}^{N} a_i x_i$$

where the coefficients, a_i, $i = 1, 2, ..., N$ are fixed.

A two's-complement representation is used for the data components which are scaled so that $|x_i| \leq 1$.

$$y = \sum_{i=1}^{N} a_i \left[-x_{i0} + \sum_{k=1}^{W_d - 1} x_{ik} 2^{-k} \right]$$

Where, x_{ik} is the k^{th} bit in $\boldsymbol{x}_i$.
W_d is the input bit width.

By interchanging the order of the two summations we get

$$y = -\sum_{i=1}^{N} a_i x_{i0} + \sum_{k=1}^{W_d - 1} \left[\sum_{i=1}^{N} a_i x_{ik} \right] 2^{-k}$$

which can be written as

$$y = -F_0(x_{10}, x_{20}, \ldots, x_{N0}) + \sum_{k=1}^{W_d - 1} F_k(x_{1k}, x_{2k}, \ldots, x_{Nk}) 2^{-k}$$

Where,

$$F_k(x_{1k}, x_{2k}, \ldots, x_{Nk}) = \sum_{i=1}^{N} a_i x_{ik}$$

F is a function of N binary variables, the i^{th} variable being the k^{th} bit in the data x_i.
Since F_k can take on only a finite number of values, 2^N, it can be computed and stored in a look-up table. This table can be implemented using a ROM (Read-Only Memory). Using Horner's method for evaluating a polynomial for x = 0.5, we can rewrite

$$y = -F_0(x_{10}, x_{20}, \ldots, x_{N0}) + \sum_{k=1}^{W_d - 1} F_k(x_{1k}, x_{2k}, \ldots, x_{Nk}) 2^{-k}$$

$$y = \left(\left(\ldots \left(\left(0 + F_{W_d - 1} \right) 2^{-1} + \ldots + F_2 \right) 2^{-1} + F_1 \right) 2^{-1} - F_0 \right)$$

Inputs, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$,…, $\boldsymbol{x}_N$ are shifted bit-serially out from the shift registers with the least-significant bit first. Bits x_{ik} are used as an address to the ROM storing the look-up table.
The computational time is W_d clock cycles. The word length in the ROM, W_{ROM}, depends on the F_k with the largest magnitude and the coefficient word length, W_c, and

$$W_{ROM} \leq W_c + \log_2(N)$$

The shift-accumulator must be able to add correctly the largest possible value obtained in the accumulator register and in the ROM. The largest value in the accumulator register is obtained when the largest (magnitude) value stored in the ROM is repeatedly accumulated.

$$y = \left(\left(\ldots \left(\left(0 + F_{W_d - 1} \right) 2^{-1} + \ldots + F_2 \right) 2^{-1} + F_1 \right) 2^{-1} - F_0 \right)$$

Thus, at the last clock cycle, corresponding to the sign bit, the value in register y is

$$|y| = \ldots(((0 + F_{max}) 2^{-1} + F_{max}) 2^{-1} + \ldots + F_{max}) 2^{-1} \leq F_{max}$$

Hence, the shift-accumulator must be able to add two numbers of magnitude $\leq F_{max}$. The necessary number range is ± 1. The word length in the shift-accumulator must be extended with guard bits for overflow detection. Hence, DA is bit serial in nature. It performs bit level rearrangement of MAC operation. It hides explicit multiplication by LUTs. Distributed Arithmetic unit essentially consists of adders augmented by a ROM.

LUT division method will be adopted in order to reduce the number of memory locations required for the design. A single LUT with 18 bit address width is partitioned into 3 LUTs with address width of 6 bits each. They are as follows:
1) 2 LUTs of 64, 12 bit wide locations.
2) 1 LUT of 64, 17 bit locations.

3. Implementation details

A reference model is generated using MATLAB simulink library for the desired filter specifications. The model is verified by giving a sine wave input. Then a verilog code is written for the high pass filter. The design is simulated using MODELSIM and verified using chipscope pro.

3.1 Design index of high pass filter

Order: 17
Sampling frequency: 8MHz
Cutoff frequency: 1.5MHz
β: 0.5
Pass band gain: 12dB
Stop band attenuation: 6dB
Input data bit width: 16 bits
Output data bit width: 33 bits

3.2 Filter coefficients

Coefficients for HPF of order 17 are designed for Kaiser window using FDA tool for the required specifications.

h(0)= 03cd, h(1)= 045e, h(2)= fece, h(3)= f8d2, h(4)= fafd, h(5)= 067d, h(6)= 101f, h(7)= 055d, h(8)= bb54, h(9)= 44ac, h(10)= faa3, h(11)= efe1, h(12)= f983, h(13)= 0503, h(14)= 072e, h(15)= 0132, h(16)= fba2, h(17)= fc33.

From the above coefficient values we see that the designed filter is a type 4 FIR filter. i.e., the filter is of even length and the coefficients exhibit antisymmetry.

3.3 Module partitioning of FIR filter

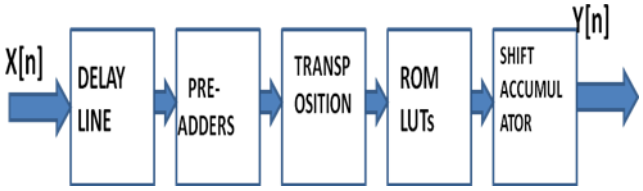

Fig 1: Module partitioning of FIR filter

Different blocks in the FIR filter module are as follows,

Delay line: The parallel input data is serialized using pipe-lined registers contained in the delay line.

Pre-adder: If the filter is symmetric, the pre-adder is implemented, that reduces by two the number of processed data after delay line.

Transposition module: Performs rearrangement of data. The rearranged data contains bits that have similar weights, but coming from different samples.ROM LUTs: Contains the pre-computed inner product terms.Shift accumulator: Calculates the sum of products.

4. Results

4.1 Impulse response

It contains 18 samples which represents filter coefficients.

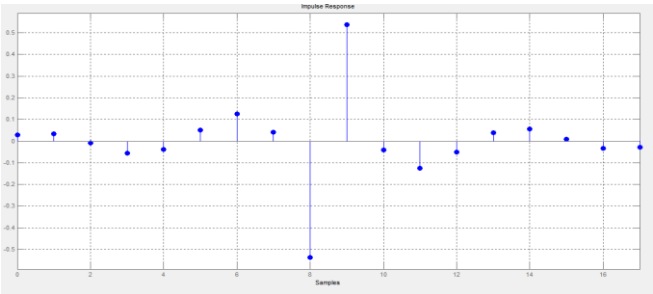

Fig **2: Impulse response of HPF of order 17**

4.2 Simulink result

The reference module generated using simulink library is verified by by giving a sine wave input of amplitude 5 and frequency 1.6MHz. Sampled at the rate of 196 samples per µs.

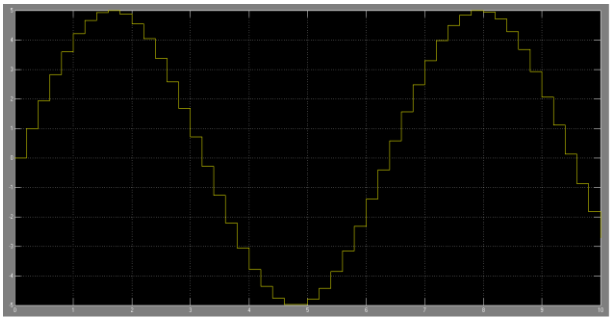

Fig 3: Sine wave input

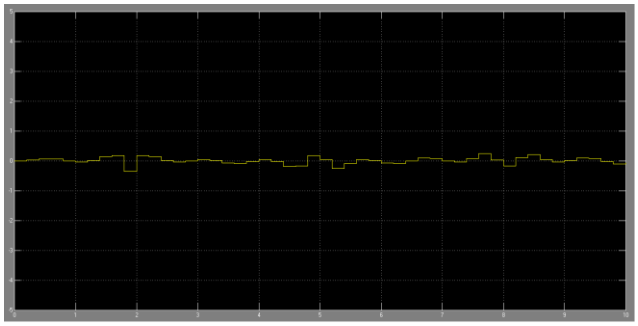

Fig 4: Output of HPF

4.3 Simulation and verification results

The final result will be obtained after 16*17= 272 clock pulses in case of DA based filter. In case of fully parallel architecture output will be obtained after 17 clock pulses, as the filter is of order 17 i.e., 17 delay elements.

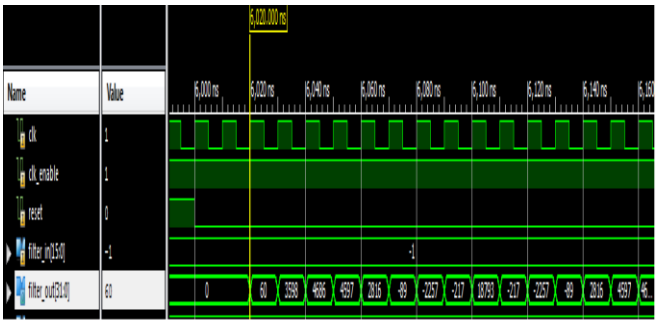

Fig 5: ISim simulation result using fully parallel architecture.

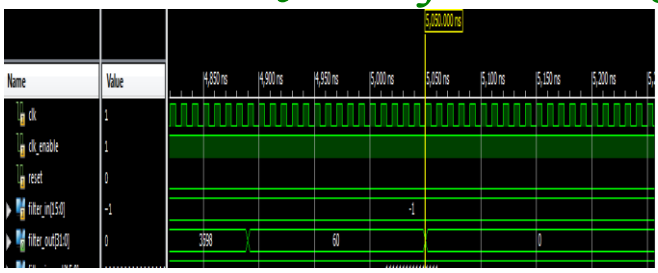

Fig 6: ISim simulation result using DA architecture

4.3 Design summary

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	320	69120	0%
Number of Slice LUTs	32	69120	0%
Number of fully used LUT-FF pairs	0	352	0%
Number of bonded IOBs	51	640	7%
Number of BUFG/BUFGCTRLs	1	32	3%
Number of DSP48Es	26	64	40%

Table1: Design summary of HP filter of order 17 with fully parallel architecture

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	166	69120	0%
Number of Slice LUTs	201	69120	0%
Number of fully used LUT-FF pairs	104	263	39%
Number of bonded IOBs	51	640	7%
Number of BUFG/BUFGCTRLs	1	32	3%

Table2: Design summary of HP filter of order 17 with distributed arithmetic architecture

DA-based filter requires less area when compared to its MAC counterparts because it don't require multipliers. However, it mandate faster clocking to maintain comparable throughput. This increase in switching activity can be mitigated through approximate processing, where power and arithmetic precision are parameters that can be traded of in an ad-hoc manner. DA architecture can be implemented on FPGA more efficiently as it eliminates the need of multipliers which are rather limited resource when compared to LUTs and adders.

5. Conclusion

The proposed design uses DA to reduce the number of multipliers required at the expense of additional adders. Exchanging multipliers with adders is advantageous because adders weigh less than multipliers in terms of silicon area. In addition the overhead from the increase in adders in preprocessing and post processing blocks stay fixed, not increasing along with the length of the FIR filter, whereas the number of reduced multipliers increases along with the length of the FIR filter.

An area efficient and hence less power consuming high pass FIR filter can be designed using DA and FPGA.

Reference

[1] Cui Guo-wei, Wang Feng-ying, The Implementation of FIR Low-pass Filter Based on FPGA and DA, 2013 Fourth International Conference on Intelligent Control and Information Processing.

[2] M. Keerthi, Vasujadevi Midasala.. FPGA Implementation Of Distributed Arithmetic For FIR Filter, International Journal of Engineering Research & Technology (IJERT)

[3] Sudhakar, Murthy.. Area Efficient Pipelined Architecture For Realization of FIR Filter Using Distributed Arithmetic, 2012 International Conference on Industrial and Intelligent Information (ICIII 2012).

[4] P.Sravanthi, CH.Srinivasa Rao, S.Madhava Rao, A Novel Approach of Area-Efficient FIR Filter Design Using Distributed Arithmetic with Decomposed LUT, IOSR Journal of Electronics and Communication Engineering (IOSR-JECE).

[5] Ankit Jairath, Sunil Kumar Shah, Amit jain, Design & implementation of FPGA based digital filters, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 1, Issue 7, September 2012.

[6] Magatha Nayak Bhukya., The Design of High Speed FIR Filter using Improved DA Algorithm and it's FPGA Implementation, International Journal of Engineering Trends and Technology- Volume3Issue2- 2012.

[7] G.B.S.R Naidu, B.Anil Kumar, G R Locharla International Journal of Scientific and Research Publications, Volume 2, Issue 7, July 2012 1ISSN 2250-3153

Biographies

1. Kavya Jyothi.B, received the B.E. degree in Electronics and communication Engineering from VTU, Belgaum, Karnataka, in 2012. Now perceiving M.Tech in VLSI and Embedded system in SSIT, Tumkur, Karnataka. *(kavya_jyothi@yahoo.com)*

2. Dr. K.B.Shivakumar Professor, Dept of Telecommunication Engineering, SSIT, Tumkur. Received the B.E. degree in Electronics and communication Engineering from Bangalore University in 1983. The M.E. degree in Electronics from Bangalore University in 1989. Ph.D in Image Processing from Fakir Mohan University, Orissa in 2012. *(kbssit@gmail.com)*

3. Dr. M. Z. Kurian. Ph.D, MISTE, MIEEE Professor & Head, Dept of Electronics and Communication Engineering, Sri Siddhartha Institute of technology, Tumkur, Karnataka, India. *(mzkurianvc@yahoo.com)*