Synthesis and implementation of DSP algorithms in advanced programmable architectures

Anna Kuncheva, George Angelov, Marin Hristov

Dept. of Microelectronics, ECAD Lab, Technical University of Sofia,

8 Kliment Ohridski bvld, Sofia, Bulgaria

anika@ecad.tu-sofia.bg, gva@ecad.tu-sofia.bg; mhristov@ecad.tu-sofia.bg

Abstract. Modern programmable technology enables the implementation of digital systems using dedicated embedded DSP blocks. Being very important part of DSP, there is a continuous need for efficient implementation of digital filters in FPGAs. In the paper, a discussion on the efficient implementation methodologies used in DSP applications aimed at state-of-the-art FPGAs is presented on the basis of a digital filter. A novel approach for digital decimation FIR filter synthesis and implementation in modern FPGAs for DSP is developed.

I. Introduction

In the recent years digital filters have been recognized as primary units in digital signal processing operation. With the advances in digital technology they are rapidly replacing analog filters, which were implemented by RLC components. Digital filters are typically implemented as multiply-accumulate (MAC) algorithms by means of dedicated DSP chips. When using pipeline architecture the speed of such implementation is limited by the speed of the array multiplier. Still, there are several other better choices for specific DSP implementations.

Programmable technology, however, allows increasing the performance of digital system by parallelisms of implemented algorithms. The flexible nature of programmable devices unfolds a new range of circuits that exploit their reconfigurable property. Thus, in many cases they make a good compromise between software and hardware solutions. A large variety of real-world applications exist for such hardware [1]. FPGAs are an array of programmable logic cells interconnected by matrix of wires and programmable switches. The ability to manipulate the logic at the gate level means that the designer can construct a custom processor to efficiently implement the desired function. FPGA manufacturers have for years now been extending their chips' ability to efficiently implement digital-signal processing. In advanced FPGA architectures, the basic DSP building blocks (for delay, data storage, multiplication, addition, subtraction, summation, and accumulation) are no longer built using discrete components. To tightly integrate these essential DSP components in high-end FPGAs, a well-planned DSP block is designed as part of the FPGA chip's dedicated resources.

Digital filters are a very important part of DSP and currently there is a need for their hardware implementation. The most common DSP function implemented in Xilinx FPGAs is the Finite Impulse Response filter. This paper shows how a Xilinx Virtex XC4VFX12-10FF668 FPGA (a reconfigurable device) can be used to implement a digital decimation FIR filter.

II. FIR filter techniques

Digital filters are typically used to modify attributes of a signal in the time or frequency domain trough linear convolution [2]. They are generally classified as finite impulse response (FIR) or infinite impulse response (IIR). According to the names, an FIR filter consists of a finite number of samples values, reducing the below presented convolution to a finite sum per output sample. An IIR filter requires an infinite sum to be performed. In this paper implementation of the decimation FIR filter will be discussed.

Efficient hardware implementation of filter's structure in programmable devices is possible by optimization of multipliers and adders implementation. In modern programmable structures specialized embedded blocks can be used to implement multipliers, increasing the performance of designed system. In case of programmable devices, however, direct or transposed forms are preferred for maximum speed and lowest resource utilization. This is caused by the fact that this approach enables exploitation parallelism of the algorithm.

The output of an FIR filter of order (length) N, to an input time-samples X(n), is given by finite version of convolution sum.

Equation view:

$$Y(n) = \sum_{k=0}^{k=N-1} h(k).X(n-k) \tag{1}$$

In (1) **h** is the coefficient, **X** is the input signal, **Y** is the output signal, **k** is number of coefficients – called taps, and **n** is the input sample number.

Diagram view:

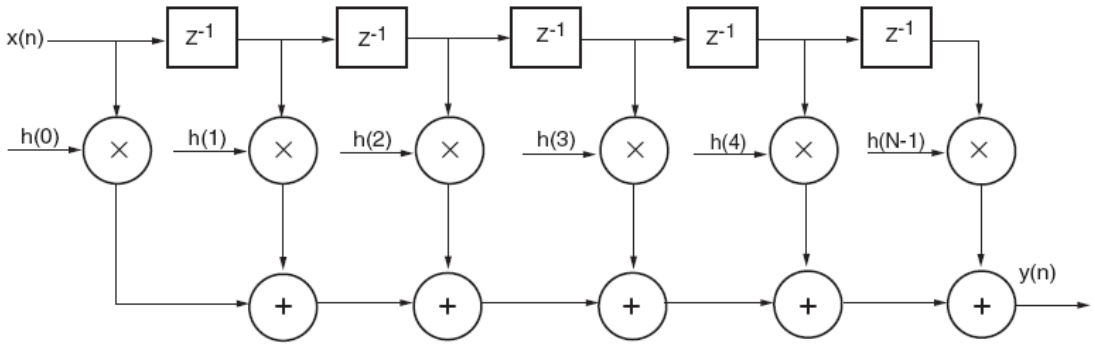

Fig. 1. FIR filter.

In result the Xilinx Virtex-4 DSP48 block supports many independent functions, including multiplier, multiplier-accumulator (MAC), multiplier followed by adder, three-input adder, barrel shifter, and pipeline. Available digital filter software allows for very easy computation of coefficients of given filer. However, the challenge is mapping of the FIR structure into suitable architecture. Digital filters are typically implemented as multiply-accumulate (MAC) algorithms with use of special DSP devices.

A. Quantization of coefficients.

Next step in this design is to select the number of bits used to represent each coefficient. It is obvious that quantization of the filter coefficient reduces the filter performance. The MATLAB tool is used to evaluate the effect that of quantization of the coefficient has on the response of each filter.

When using 24 bits for quantization, the response of the quantized filter matches closely the reference filter, both in the passband and stopband. It is shown in Figure3.and Figure4. Therefore, 24 bits for the coefficients FIR filter have been selected.

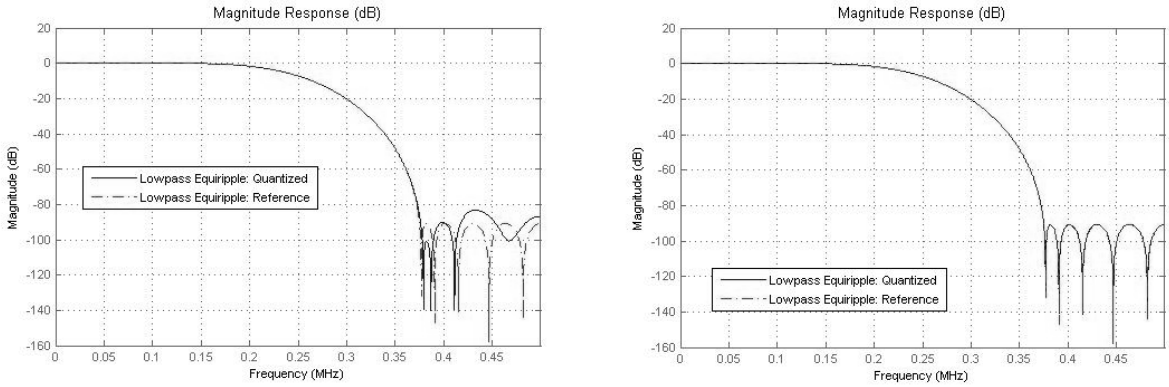

Fig. 3. Frequency response of FIR filter with and without coefficient quantization

III. Hardware implementation

A. How to implement this filter in FPGA

The goal is to implement a filter that meets all specifications, while making the most efficient use of hardware available. Since fast multipliers for data with relatively large number of bits are difficult to implement, and require a large amount of resources, it is important to try to reduce the number of multiplier required. Several methods can be used for this. A first step toward reducing the number of multiplication is making use of the fact that, for linear-phase response filter, coefficients are symmetrical. This is will half the number of multiplications required. A first step toward reducing the number of multiplication is making use of the fact that, for linear-phase response filter, coefficients are symmetrical.

The general formulae to compute the output of an FIR filter with N coefficients is given by Formula 1:

$$Y(n) = \sum_{k=0}^{k=N-1} h(k).X(n-k) \tag{1}$$

With symmetrical coefficient, and assuming an even number of coefficients, we have:

$$
\begin{aligned}
h_0 &= h_{N-1} \\
h_1 &= h_{N-2} \\
&\ldots \\
h_{N/2-1} &= h_{N/2}
\end{aligned}
\tag{2}
$$

Therefore, the output is now:

$$
y_n = \sum_{k=0}^{\frac{N}{2}-1} (x_{n-k} + x_{n-N+h+1}) \times h_k \tag{3}
$$

This is will half the number of multiplications required.

B. Efficient hardware implementation

Programmable technology allows also application of special filter techniques. The dedicated DSP blocks in high-end FPGAs – such as the Xilinx® XtremeDSP™ slice (also referred to as DSP48) in Virtex™-4 devices – are playing a critical role in designing high performance DSP systems. Below we show how to use a Xilinx Virtex XC4VFX12-10FF668 FPGA (a reconfigurable device) in order to implement a digital decimation FIR filter.

The main components used to implement a digital filter algorithm include adders, multipliers, storage, and delay elements. The DSP48 slice includes all of the above elements, making it ideal to implement digital filter functions. All of the input samples from the set of *n* samples are present at the input of each DSP48 slice. Each slice multiplies the samples with the corresponding coefficients within the DSP48 slice.

The outputs of the multipliers are combined in the cascaded adders. In Fig.1 , the sample delay logic is denoted by Z^{-1}, where the "–1" represents a single clock delay. The delayed input samples are supplied to one input of the multiplier. The coefficients (denoted by *k*0 to *k*(*N*-1)) are supplied to the other input of the multiplier through individual BRAM. *y*(*n*) is merely the summation of a set of input samples, and in time, multiplied by their respective coefficients.

The FIR filter is written in VHDL and was implemented in a Xilinx Virtex XC4VFX12-10FF668 FPGA. Value of quantize coefficients are obtained using Matlab. Afterwards, a program written in VHDL converts the coefficients to a constant array. BRAM are used to store coefficient values and data samples.

Number of coefficients, coefficients values, coefficients and input word length and decimation factor are all defined in the package file. This FIR filter must have an even number of coefficient, and uses symmetry to reduce number of multiplication by 2. It can decimate by any integer; for no decimation, set the decimation factor to one.

There are three distinct operations performed by the FIR filter:

1. Adds the data corresponding to symmetrical coefficient, and fetch the coefficient.
2. Multiplies the sum of the two data by coefficient.
3. Accumulates the result of multiplication, until all coefficients are multiplied, at which point the content of the accumulator is written on the output and its value is reset.

Arithmetic is performed using two's complement:

- Data is converted to positive value before multiplication, and its sign is registered.

- Coefficients are stored using their positive value, with MSB being the sign bit, which is registered before multiplication
- Multiplication is done with positive data and coefficient, and the proper sign restored afterwards depending on the registered sign of both data and coefficient.
- A crucial part of the design is the generation of the timing signal inside the *fir_timing* module. This creates a signal timer used to fetch coefficient and data from the registers. It also creates a run signal that is used mainly by the accumulator to stop it from accumulating when all calculations are done.
- Data samples are stored in RAM (component *data_register*)
- To decrease latency of filter the multiplication of all previous data samples is first performed, so that when a new data is received there is only one more multiplication to do before the output is computed.
- Latency of the filter is 4 clock cycles

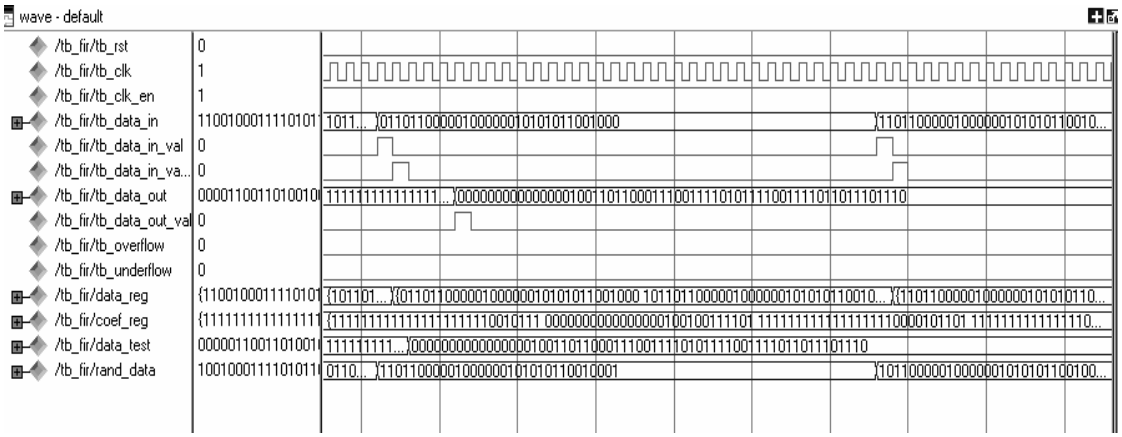

Fig. 2. FIR Filter simulation waveforms.

A clock enable signal has been added for debugging purpose and could be used to resynchronize asynchronous data with the system clock. Two signals indicate whether the results from the accumulator are overflowed or underflowed (by one bit only). Those signals are only valid for one clock cycle (same cycle as *data_out_val*)

Tab. I. Implementation results		
Number of Slice Flip Flops	236 out of 10994	2%
Number 4 input LUTs	424 out of 10994	3%
Number of FIFO16/RAMB16s	2 out of 36	5%
Number of bonded IOBs	99 out of 320	30%
Number of DSP48s:	4 out of 32	12%

IV. Conclusion

The simple, traditional adder-tree approach limits the performance and extensibility of a given filter implementation. By using adder-chain-style implementations (Virtext-4 FPGA enables this implementation approach because of the DSP blocks embedded in the chip), these limitations are overcome. Using mathematical computation packages such as Matlab, makes it possible to quickly find the filer order, the required quantization level for the coefficients as well as their values. Analyzing the design we have found an efficient way for the hardware filter implementation. The use of top-down design methodology decreases the total design time. Finally, the high level hardware description language VHDL fully supports arithmetic and binary manipulations that are specific for this design.

References

[1] Villasenor J. and W. Mangione-Smit (1997) *"Configurable computing"*. Scientific American,

[2] Vinger. K, Jim Torresen (2003) Implementing Evolution of FIR-Filters Efficiently in an FPGA. *NASA/DoD Conference on Evolvable Hardware (EH-2003)*, Chicago, Illinois, USA

[3] Xilinx, (Dcember 2005) XtremeDSP for Virtex-4 FPGAs Xilinx, User Guide.

[4] Kuncheva A.S., T. Mougel, L. Fujcik, B. Donchev, M.H. Hristov,(2005) Design of decimation filter for novel sigma-delta modulator, *The 14th International Scientific and Applied Science Conference - Electronics 2005,* Book 5, pp. 56-61, ISBN 954 438 521