DESIGN AND IMPLEMENTATION FIR FILTERS USING FPGA

SŁAWOMIR JASTRZĘBSKI

University of Technology and Life Sciences
ul. Kordeckiego 20, 85-225 Bydgoszcz, Poland
sj@utp.edu.pl

Underwater acoustic channels are characterized by multipath phenomenon whose characteristics are time varying. Multipath propagation contributes to signal fading, and causes intersymbol interference (ISI) in a digital communication system.

Raised cosine filters are widely used in wireless communication systems and the effects of these filters are crucial to wireless communication systems, for example underwater communication systems. Pulse shaping filters are commonly used in digital data communication systems to limit intersymbol interference. Thus, digital filters have been recognized as primary digital signal processing operation. In order to apply DSP algorithms to wireless communication systems high density Field Programmable Gate Arrays have recently emerged as ideal implementation platforms for digital filters due to its potential speed and flexibility.

This paper presents the design and implementation of FIR filters using FPGA technology. The following architectures of filters are studied: multiply and accumulate (MAC) standard FIR filter, parallel transposed FIR filter, and direct-form filter using Distributed Arithmetic (DA). The proposed filters have been designed and synthesized with ISE software, and implemented with a Virtex-II FPGA device.

INTRODUCTION

Many digital systems use signal filtering to remove unwanted noise to provide spectral shaping, or to perform signal detection or analysis. Filters are used with communication applications such as band selection and low-pass filtering. Two types of filters that provide these functions are FIR and IIR filters. FIR filter is the foundational element of DSP. FIR filters can be used in systems that require a linear phase and have an inherently stable structure.

In digital communication, matched filters are widely used in wireless communication systems and the effects of these matched filters are crucial to wireless communication systems. In order to improve the performance the matched filter must be designed in digital methods. Because of FIR filters' linear phase characteristics, they are widely used in matched filters design. The amplitude frequency response of a FIR filter can match that of a matched filter in precise accuracy, which will decrease the introduced noise and improve the performance of the wireless communication system. However, the working speed of the digital FIR matched filters is often relatively lower than other components in the wireless communication system and it is relatively difficult for the wireless communication system with such digital FIR filters to work at a high sampling rate. In digital underwater communication, pulse-shaping filters allow the transmission of pulses with negligible intersymbol interference. Therefore, these filters must have a frequency response with sufficient selectivity and attenuation to suppress noise and interference in adjacent channels.

Recently, a large number of researchers have focused on their interests on the FIR filters and their implementation in FPGAs, which are increasingly becoming the implementation platform of choice for high-speed DSP systems. Progress in development of programmable architectures observed in recent years resulted in digital devices that allow building very complex digital circuits and systems at relatively low cost in a single programmable structure. Programmable technology provides possibility to increase the performance of digital system by implementation of multiple, parallel modules in one chip. Due to the advantages of versatility, flexibility, large scale of FPGA, a number of applications in digital system designing based on FPGA have been developed for telecommunications, controlling and information processing system. FIR digital filters were modeled with VHDL coding and synthesized using Xilinx Synthesis Technology (XST) software and then implemented on Xilinx Virtex-II FPGA chip using Xilinx ISE Foundation. The functional simulations and post-timing simulations were performed in the verification step using Modelsim software.

1. FIR FILTER ARCHITECTURES

The structure of a FIR filter is a weighted, tapped delay line. The filter design process involves identifying coefficients that match the frequency response specified for the system. The coefficients determine the response of the filter. By changing the coefficient values or by adding more coefficients to the filter, we can change the signal frequencies, which pass through the filter. A discrete-time linear finite impulse response filter generates the output $y[n]$ as a sum of delayed and scaled input samples $x[n]$. This process is formally described by following formula:

$$y[n] = \sum_{k=0}^{K-1} h[k]x[n-k] \tag{1}$$

where $h[k]$ are filter's coefficients.

A direct implementation requires K multiply and accumulates (MAC) operations, which are expensive to implement in hardware due to logic complexity and area usage. Fig. 1 shows the direct form FIR filter.

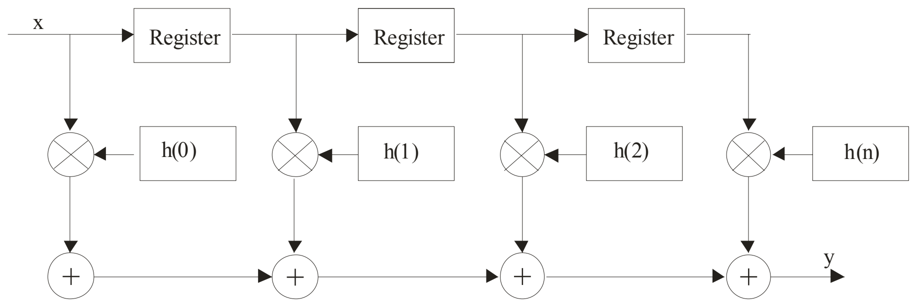

Fig.1 Direct form FIR filter

Multiplications are the critical operation in any DSP algorithm. Bit-serial and serial-parallel techniques provide a convenient trade-off between size and speed of a multiplier. When speed of operation is a design issue, parallel array multipliers have to be considered. FIR filters can be implemented in different structures. The following architectures of filters are studied in this section: multiply and accumulate (MAC) standard FIR filter, parallel transposed FIR filter, and direct-form FIR filter using Distributed Arithmetic. Typically, the FIR filter function is based on a MAC operation.

1.1 MAC UNIT REALIZATION

The base of many DSP algorithms is a MAC function. The multiply accumulate block is comprised of a multiplier and accumulator block. The multiplier computes the product of a filter tap and a sample from the data buffer, and the accumulator computes a running sum of these products. MAC function uses a conventional sequential shift-and-add technique to multiply one number by another and sum the results (Fig. 2). The multiplier can be implemented either in the different form of RTL structures or using embedded multipliers.

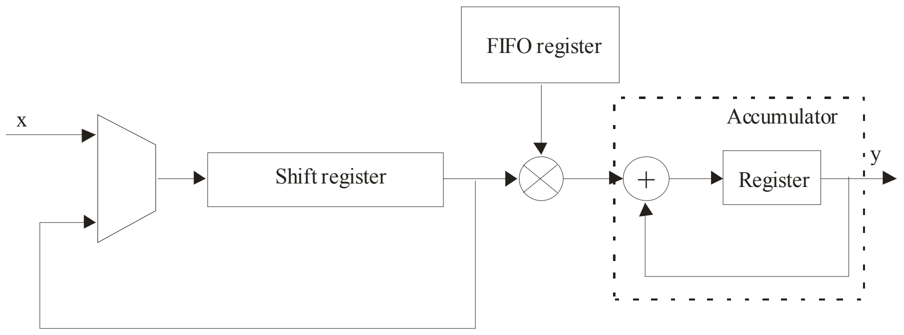

Fig.2 Direct form FIR filter using single MAC block

The accumulator is configured to reinitialize upon reset to its current input value to avoid a one-clock cycle stall at the end of each sum-of-products computation. A register captures the output of the MAC engine before it is reset. Alternatively, the MAC operations may be replaced by a series of look-up-table (LUT) accesses and summations, known as Distributed Arithmetic [1], [2], [3].

1.2. DISTRIBUTED ARITHMETIC REALIZATION

DA is a well-known method to save resources in MAC structures utilized to implement DSP functions. In many DSP applications, a general-purpose multiplication is not required. DA is a bit-serial operation that implements a series of fixed-point MAC operations in a fixed number of steps, regardless of the number of terms to be calculated. This arithmetic trades memory for combinatory elements, resulting ideal to implement custom DSP in LUT-based FPGAs. In a DA bit-serial implementation of a FIR filter, each product term is addressed once per bit. After the last product-term has been obtained, it is added with its appropriate shift with the rest of the product term previously added. The structure that represents the 4-tap FIR filter using Distributed Arithmetic is showed in Fig. 3.

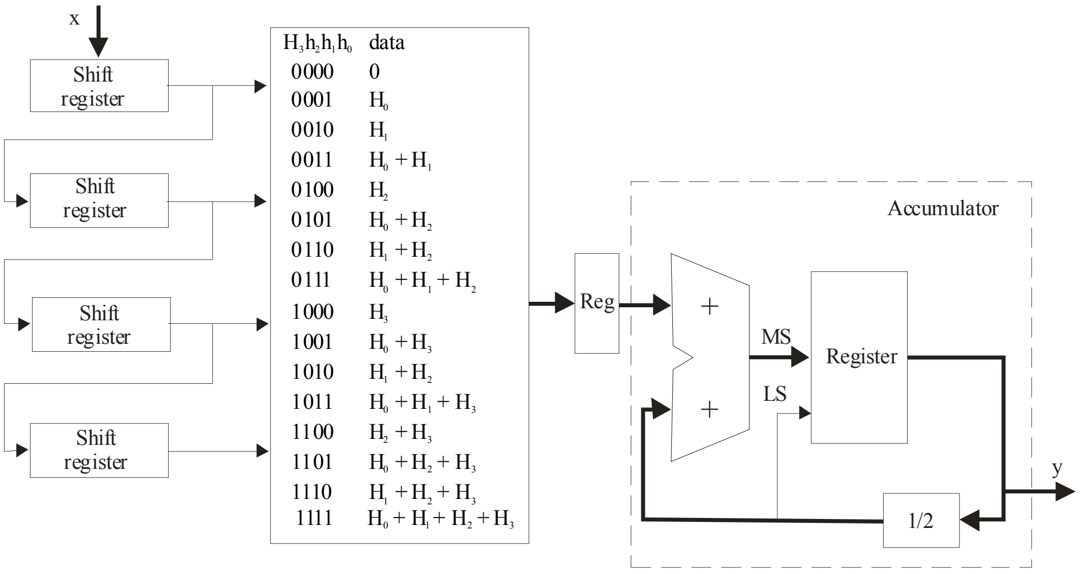

Fig.3 LUT-based DA implementation of a 4-tap FIR filter

1.3 TRANSPOSED FORM FIR FILTER

Another form of FIR filter is structure called the transposed FIR filter (Fig. 4). The transposed FIR yields an identical mathematical response but with several advantages for FPGA implementation. This filter utilizing the same resources, but data samples are applied in parallel to all the tap multipliers. The input registers are not required, because high fan-out input signals can be handled by the Virtex-II architectures. In the transposed FIR filter, identical tap coefficient magnitudes can share multiplication hardware because taps receive the input sample simultaneously. The products are applied to a cascaded chain of registered adders, combining the effect of accumulators and registers. The order of tap coefficients must be reversed with the first tap closest to the output. Traditional FIR filters are implemented in dedicated hardware without any parallelism, thus limiting the sample rate. The Virtex-II FPGAs have abundant hardware resources to facilitate full parallelism. In a parallel implementation of a filter, each tap has a dedicated multiplier.

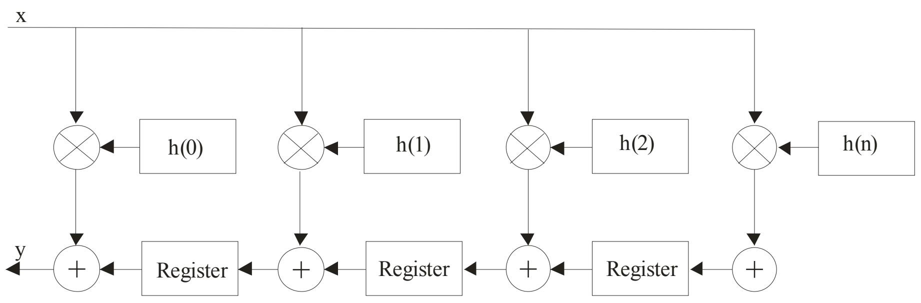

Fig.4 Transposed form FIR filter

2. FPGA IMPLEMENTATION RESULTS

The described FIR filter models have been implemented in very high speed integrated circuit (VHSIC) hardware description language (VHDL). VHDL is a kind of language for describing VHSIC in IEEE standards. It has been widely used in deigning VHSIC due to its distinct advantages. The FIR filters were implemented on RTL and structural level. The model supports generic parameters so that word length and filter length can be adjusted to the application requirements. All implemented filters have 16 taps, 16-bits inputs, 16-bit signed coefficients, and were targeted to Virtex-II XC2V1000 device.

First implementation of FIR filter was serial structure consists of a shift register, multiplier and an adder. This serial MAC form filter was shown in Fig. 5.

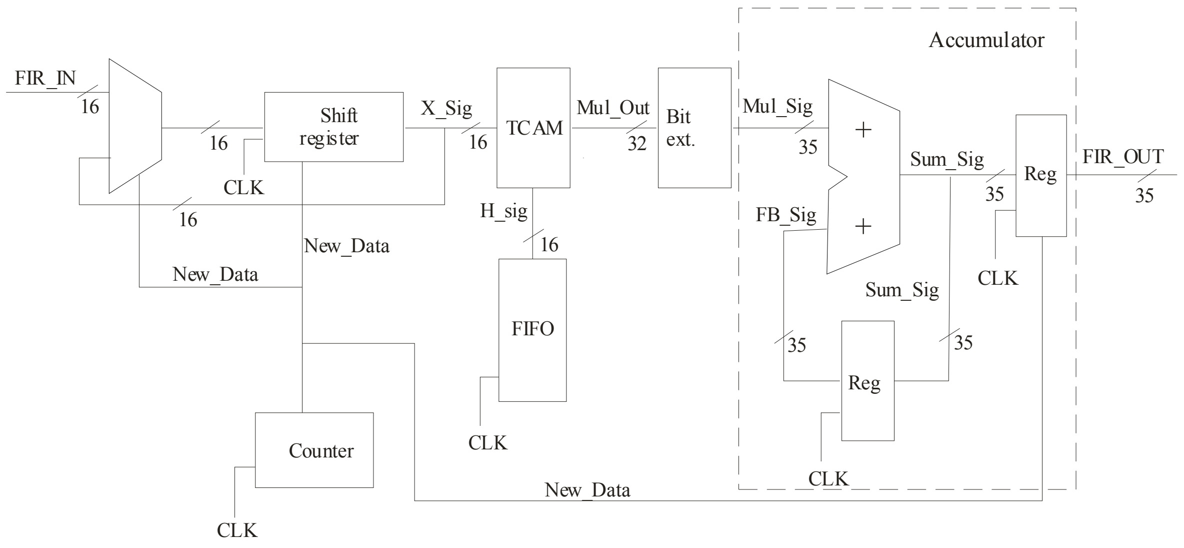

Fig.5 Block structure of direct form FIR filter using single MAC block

All these elements were implemented in the Xilinx FPGA Virtex-II device. Multipliers and adders are the basic components of all digital filters. For multiplier performance improvement, the features of the filter have to be carefully studied. The efficiency of the multiplier determines the overall performance of the filter. Hence, the multiplier must be implemented for the best possible performance. In this implementation, traditional two-variable multiplier (TCAM) [4] and carry-lookahead adder [5] were used. Both multiplier and adder were modeled in VHDL on RTL and structural level. The FIR filter coefficients are stored in FIFO register. The sequential use of this single MAC engine limits the sample rate of an FIR filter, as each tap will require a MAC instruction to be performed.

If very high sampling rates are required, full-parallel hardware must be used where every clock edge feeds a new input sample and produces a new output sample. In this paper such filter was implemented on FPGA using transposed FIR filter structure. Fig. 6 shows full-parallel, fixed-coefficient transposed FIR filter structure which was realized in VHDL [6].

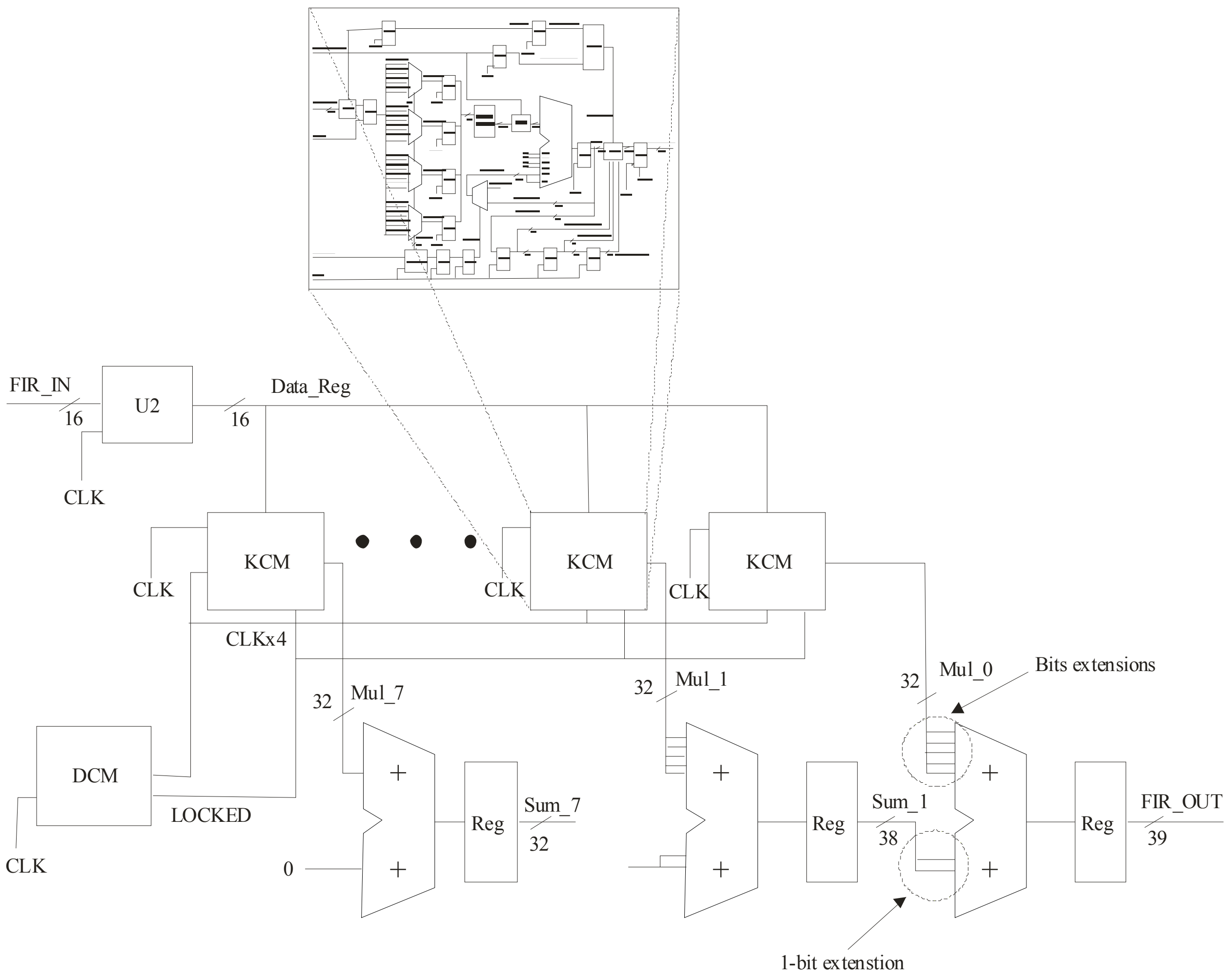

Fig.6 Block structure of transposed form FIR filter (parallel implementation)

The tap data is an input of multiplier, the other a constant coefficient. Since one input is a constant, these multipliers are called Constant Coefficient Multipliers (KCM) [7], [8], [9]. Constant Coefficient Multiplier was implemented in FPGA device using the structural technique. In this case LUT memory, store 16 partial products relating to the fixed coefficient and then use a simple adder to combine these products. KCM’s are less than one third of the size of traditional two-variable multiplier (Table 1). Fig. 7 shows the simple realization of KCM multiplier, but pipelining and resource sharing of adders further enhance the performance of KCM structure. Fig. 8 shows such multiplier block and this multiplier was used in fully-parallel transposed form of FIR filter. Table 1 shows the results of logic slice utilization (Virtex-II XC2V1000 device) for three different structures of multipliers. The FIR filter was built by utilizing the KCM blocks, delay elements, adders and Digital Clock Manager (DCM) component. The FIR filter coefficients are stored in ROM-based LUT.

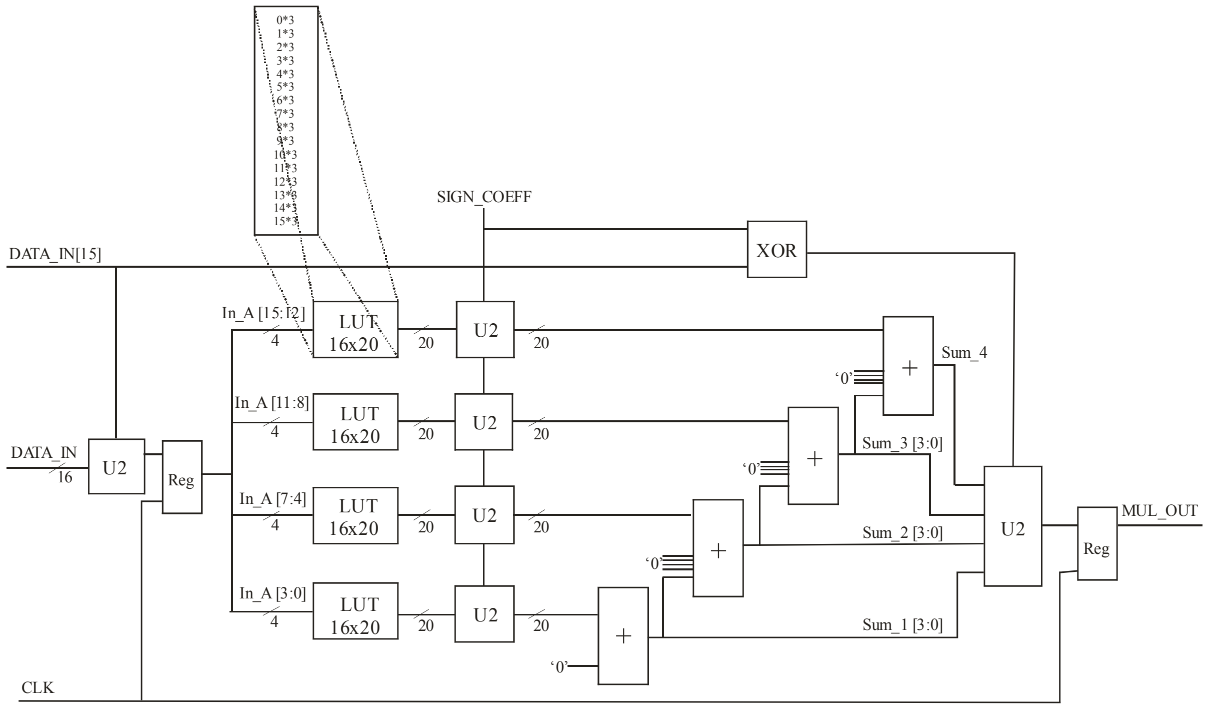

Fig.7 Constant Coefficient Multiplier block diagram

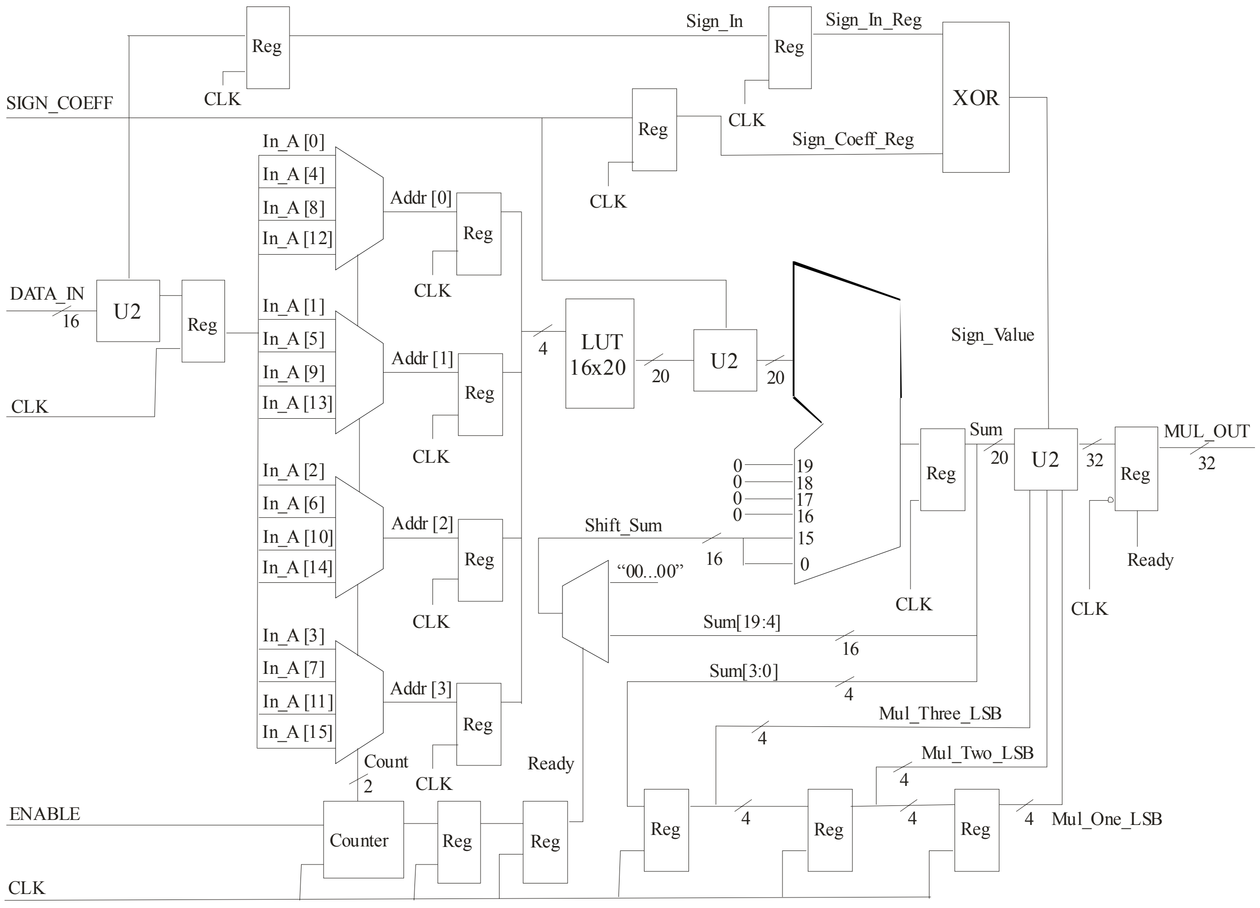

Fig.8 Constant Coefficient Multiplier block diagram using in transposed form FIR filter

Tab.1 Virtex II (XC2V1000) logic slice utilization for three multipliers configuration

	Maximum	TCAM [4]		KCM (Fig. 7)		KCM (Fig. 8)	
		#	%	#	%	#	%
SLICES	5120	253	4	158	3	106	2

The FPGA devices have the ability to implement a FIR filter function using Distributed Arithmetic method. This technique was used to optimize the implementation of MAC-based algorithm. The architecture of this filter was shown in Fig. 9. The FIR filter has one long bit-wide shift register that taps into each word. The input data is loaded parallel into 16-bit register that can shift its contents to the next register. A chain of 16 such registers was provided. Only the first 16-bit word is parallel loaded. The shift register then serially shifts out all the data-bits, 1-bit at a time. The output bit from the shift register addresses 1-bit of a corresponding LUT, until the most significant bit of n-bits has been clocked out. When the shifting is complete, the first data shift register is empty and is ready to be parallel loaded with the next word and the process repeats. This 16-tap FIR filter requires a 16-bit LUT. The optimum partition for four-input function generators was four products per LUT. In this project four 4-bit (16-word) LUT memories were used. The compilation results in the Xilinx ISE software show that the maximum clock frequency of the filter could be over 104MHz. The detailed compilation results are listed in a Table 2. SDA techniques results in area efficient design of an FIR filter.

Tab.2 Virtex-II (XC2V1000) logic slice utilization and performance for different FIR filter structures

FIR Filter	SLICES			Clock frequency
	Maximum	Used		
		#	%	MHz
MAC	5120	545	10	45
Transposed form	5120	614	11	24
SDA	5120	177	3	104

In a serial MAC based FIR realization, the sample throughput is coupled to the filter length. The filter sample throughput is inversely proportional to the number of filter taps. As the filter length is increased, the system sample rate is proportionately decreased. This is not the case with serial DA based architectures. With DA architecture, the system sample rate is related to the bit precision of the input data samples. For *n*-bit precision input samples, *n* clock cycles are required to form a new output sample. The bit-clock frequency is greater than the filter sample rate f_s and is equal to nf_s. The filter sample rate is not coupled from the filter length. As the filter length is increased in a DA FIR filter, more logic resources are consumed, but throughput is maintained. The performance for the 16-tap FIR filter implemented with a serial direct form MAC algorithm, resulted in clock frequency about 45MHz, in 545 slices. The filter sample rate is 45MHz/16taps=2,8MHz. A parallel transposed 16-tap FIR filter, as shown in Figure 6, resulted in a sample rate of 24MHz, in 614 slices. The filter design implemented in an FPGA with 16-bit serial DA resulted in more than twice the performance (104MHz/16bits=6,5MHz) compared to the direct form MAC algorithm. Transposed FIR filter uses a much larger number of slices then SDA filter, but processing all data samples in parallel, at the data sample rate (24MHz). This filter is the fastest among all implemented FIR filters.

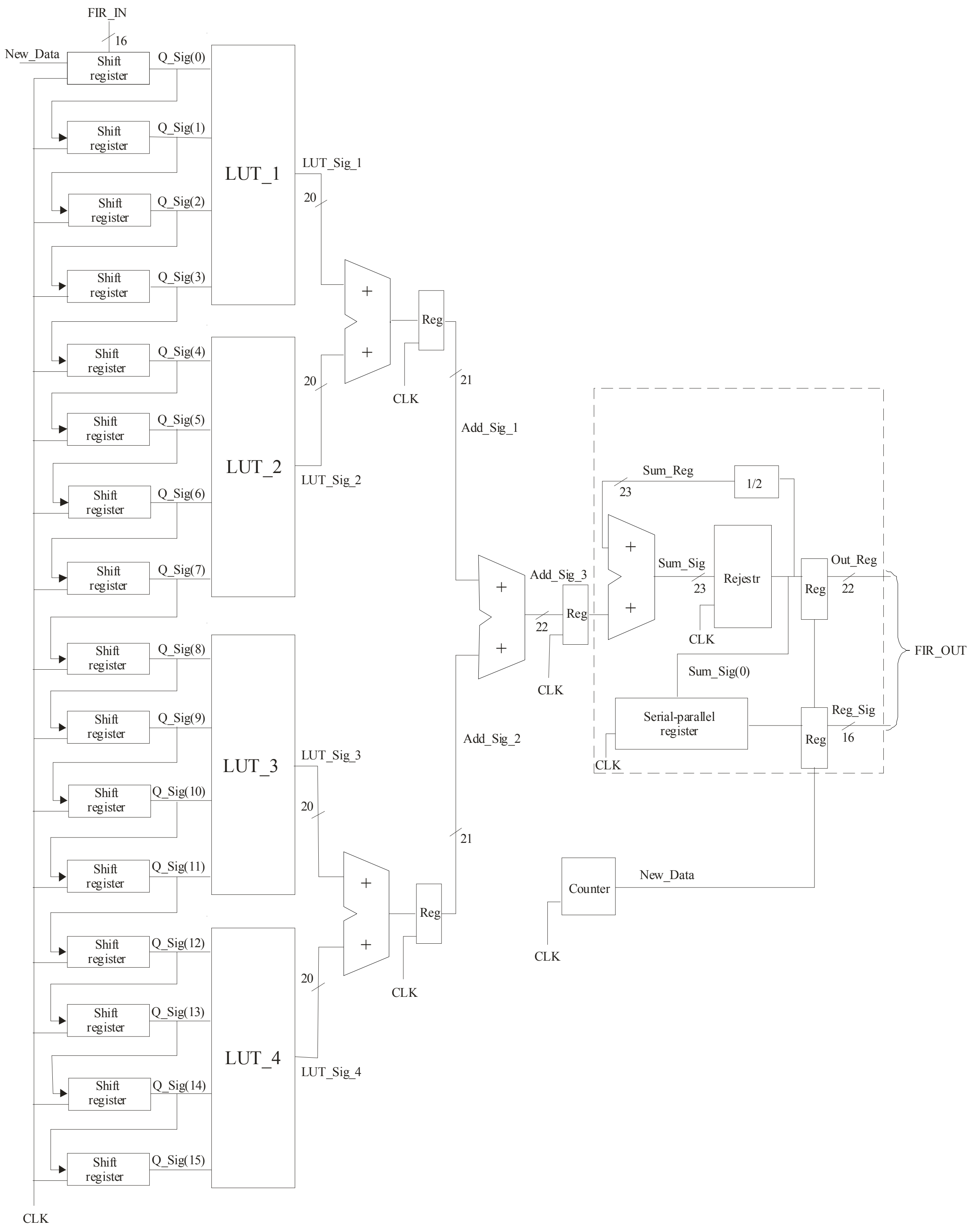

Fig.9 Block structure of serial DA implementation of a 16-tap FIR filter

3. CONCLUSIONS

The FIR filters implemented in Virtex-II FPGA provide the designer tremendous flexibility in terms of the number of filter taps and changes in existing coefficients. This paper has described how FIR filters can be implemented on FPGA devices. We can use different techniques to implement such filters. The FIR filters implemented were tabulated for reference and guidance in Table 2. It has been shown that SDA technique provides the best trade-off between speed and resource requirements and is very efficient to implement. Synthesis results have indicated that a 16-tap parallel transposed filter operating at 24 MHz can be realized. It is expected that the sample rate can be further increased by optimizing placement and routing. The MAC function can be implemented more efficiently with Distributed Arithmetic techniques then with conventional arithmetic methods. DA can make extensive use of look-up tables, which makes it ideal for implementing DSP functions in LUT-based FPGAs and this technique exhibits efficiency in terms of area. Parallel implementation shows a very good sample rate. This technique is the best choice for implementing FIR filters w FPGAs. Future work will be concentrated on experiments with parallel Distributed Arithmetic (PDA) to increase the overall performance of serial Distributed Arithmetic.

REFERENCES

1. M.M. Eshtawie, M. Othman, Distributed Arithmetic Implementation of an Optimized Raised Cosine FIR Filter Coefficients, ICSP2006 Proceedings, vol.1, pp. 16-20, 2006.
2. S. Wang, B. Tang, J. Zhu, Distributed Arithmetic for FIR Filter Design on FPGA, ICCCAS 2007**,** pp. 620-623, 2007.
3. G.R. Goslin, A Guide to Using Field Programmable Gate Arrays (FPGAs) for Application-Specific Digital Signal Processing Performance, v.1.0, Xilinx, 1995.
4. M.J. Schulte, P.I. Balzola, A. Akkas, R.W. Brocato, Integer Multiplication with Overflow Detection or Saturation, IEEE Transactions on Computers, vol.49, pp. 681-691, 2000.
5. F.C. Cheng, S.H. Unger, M. Theobald, Self-Timed Carry-Lookahead Adders, IEEE Transactions on Computers, vol.49, pp. 659 – 672, 2000.
6. V. Pasham, A. Miller, K. Chapman, Transposed Form Filters, Xilinx Applicaton Note, 2001.
7. K. Wiatr, E. Jamro, Układy mnożące przez stały współczynnik implementowane w układach programowalnych FPGA, Kwartalnik Elektroniki i Telekomunikacji 2/2001, ss. 233-253, 2001.
8. K. Wiatr, E. Jamro, Implementacja szybkich układów mnożących w strukturach FPGA, Kwartalnik Elektroniki i Telekomunikacji 4/2001, ss. 495-514, 2001.
9. K. Chapman, Constant Coefficient Multipliers for the XC4000E, Xilinx Applicaton Note, 1996.