ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

The Design and Implementation of Convolution into FPGA

Banoth Kranthi Kumar
Assistant Professor, M.Tech, Vidya Bharathi Institute of Technology, India
kranthi123vbit@gmail.com

Abstract

We present original approach to the design and implementation of the convolution in this article. This paper presents a direct method of reducing convolution processing time using hardware computing and implementations of discrete linear convolution of two finite length sequences (NXN). This implementation method is realized by simplifying the convolution building blocks. The purpose of this research is to prove the feasibility of an application specific integrated circuit (ASIC) that performs a convolution on an acquired image in real time. The proposed implementation uses a modified hierarchical design approach, which efficiently and accurately speeds up computation; reduces power, hardware resources, and area significantly. The efficiency of the proposed convolution circuit is tested by embedding it in a top level FPGA.

Simulation and comparison to different design approaches show that the circuit uses only 5mw that saves almost 45% of area and is four times faster than what is implemented in. In addition, the presented circuit uses less power consumption and has a delay of 10ns from input to output using 32nm process library. It also provides the necessary modularity, expandability, and regularity to form different convolutions for any number of bits.

Keywords: ASIC, Convolution.

Introduction

Convolution is a mathematical way of combining two signals to form a third signal. It is the single most important technique in Digital Signal Processing. Using the strategy of impulse decomposition, systems are described by a signal called the impulse response Convolution I important because it relates the three signals of interest: the input signal, the output signal, and the impulse response. This chapter presents convolution from two different viewpoints, called the input side algorithm and the output side algorithm. Convolution provides the mathematical framework for DSP. Many image processing operations such as scaling and rotation require re-sampling or convolution filtering for each pixel in the image. Convolutions on digital images are important since they represent operations that are more general than the operations that can be performed on analog images. Digital images can be modified (through convolution) by neighborhood operations; these operations go beyond point wise operations, and include smoothing, sharpening, and edge detection. Convolution has many applications which have great significance in discrete signal processing. It is usually difficult to deal with analog signals. Hence signals are converted to digital state. Filtering of signals is very important in order to determine which one to accept and which one to reject, and all of that is done by convolution. Some of the major uses of convolution are state Image processing; Wavelets generated by using discrete singular convolution kernels and Fourier transform applications . Many approaches have been attempted to reduce the convolution processing time using hardware and software algorithms. But they are restricted to specific applications Presented a design for fast convolve for CDMA signals. This is based on avoiding complex operations such as FFT based convolves. They used substitution of the FFT for a Walsh which reduces the operations three times because it uses only real additions but it requires more hardware like counters, and RAM blocks which increases activity factor. Using image processing functions such as convolution filtering, high performance can be achieved by exploiting parallelism and minimizing hardware cost, but different filter widths and thus potentially different hardware structures are needed for different applications. It is therefore difficult to make a fixed parallel structure efficient. In an application involving spatial scaling of images, for example, a larger filter kernel would be required for large scale factors, a small one for modest scaling. It would be

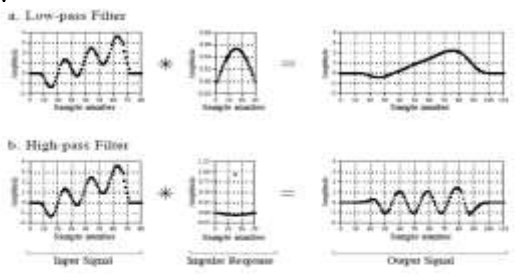
ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

expensive to implement the entire largest desired filter kernel, and wasteful for small scale factors. It is proven that convolution can check all the phase shifts in one step. This is usually done by using the known FFT-based convolution . Each FFT (or IFFT) requires N*log*N complex multiplications and N*log*N complex additions. Therefore, some algorithm require approximately 3N(*log*N)+N complex multiplications and 3N(*log*N)+N additions. Implementing the algorithm in parallel hardware will speed up the process but the implementation itself is very complex and requires a huge silicon area. The main problem in implementing and computing convolution is speed, area and power which affect any DSP system. Speeding up convolution using a Hardware Description Language for design entry not only increases (improves) the level of abstraction, but also opens new possibilities for using programmable devices. Today, most DSPs suffer from limitations in available address space, or the ability to interface with surrounding systems. The use of high speed FPGAs, together with DSPs, can often increase the system bandwidth, by providing additional functionality to the general purpose DSPs.In this paper, a novel method for computing the linear convolution of two finite length sequences is presented. A 4x4 convolution circuit can be instantiated for larger ones. This method is similar to the multiplication of two decimal numbers, this similarity that makes this method easy to learn and quick to compute .This paper is organized as follows. Section II investigates the related convolution algorithm implementation. In section III, circuit implementations are presented. Section IV presents the verification of the proposed design. In section V, evaluation and comparison of the design are presented. Finally, the conclusion is obtained. The impulse response goes by a different name in some applications. If the system being considered is a filter, the impulse response is called the filter kernel, the convolution kernel , or simply, the kernel . In image processing, the impulse response is called the point spread function. While these terms are used in slightly different ways, they all mean the same thing, the signal produced by a system.

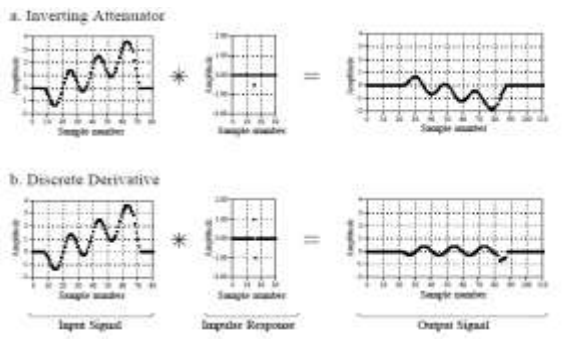
Background Work

Convolution being used for low-pass and high-pass filtering. The example input signal is the sum of two components: three cycles of a sine wave (representing a high frequency), plus a slowly rising ramp (composed of low frequencies). In (a), the impulse response for the low-pass filter is a smooth arch, resulting in only the slowly changing ramp waveform being passed to the output. Similarly, the high-pass filter, (b), allows only the more rapidly changing sinusoid to pass. Figure 1.1 illustrates two additional examples of how convolution is used to process signals. The inverting attenuator, (a), flips the signal top-for-bottom, and reduces its amplitude. The discrete derivative (also called the first difference), shown in (b), results in an output signal related to the *slope* of the input signal. Notice the lengths of the signals in Figs. 1.1 and 1.2. The input signals are 81 samples long, while each impulse response is composed of 31 samples. In most DSP applications, the input signal is hundreds, thousands, or even millions of samples in length. The impulse response is usually much shorter, say, a few points to a few hundred points. The mathematics behind convolution doesn't restrict how long these signals are. It does, however, specify the length of the output signal. The length of the output signal is equal to the length of the input signal, plus the length of the impulse response, minus one.

.



For the signals in Figs.1.1 and 1.2, each output signal is: 81% 31& 1 ’ 111 samples long. The input signal runs from sample 0 to 80, the impulse response from sample 0 to 30, and the output signal from sample 0 to 110

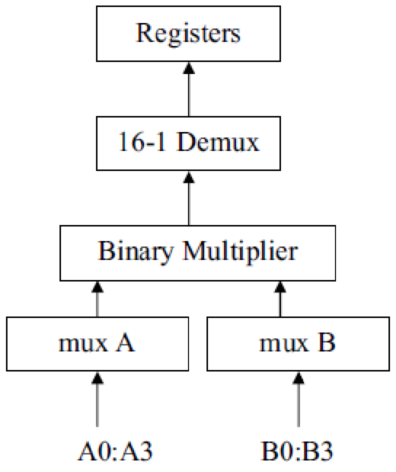


Proposed Implementation Circuit

NXN was selected, and the implementation for 4x4 was prepared in order to have short convolutions that will lead to the lowest

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

implementation cost, as mentioned in . The circuit deals with two signals having N values each. We selected N=4 in our implementations. We consider the two numbers like two arrays having four locations each to store values. Each array is fed into a quadruple 4X1 Mux separately. Hence we can have each signal value up to 4 bit. The selection of values is done by selection switches of each Mux. The selected values go into the Array Multiplier and from there they are routed into Parallel Load Registers through a 1X16 Demux. Afterwards the stored values are added to get the convolved Result [4]. The block diagram of the circuit is shown in figure



The basic concept of convolution is to flip, multiply and add. Now for two signals of four values each, we have to flip (invert one of the signals) multiply and then add the values. The flipping of the values is done by selection of the 4X1 Multiplexer. Figure 4 shows the basic building blocks used in the design. The design is built in Verilog and implemented on an FPGA. We parameterized the inputs to N, so we can setup the values to whatever number we need. Further, a 16X1 Demultiplexer is used to store the multiplied values in different registers. The values of the first and seventh registers are first and seventh output values respectively. The other values are obtained by adding the corresponding values. For addition an 8-bit Full Adder was made by instantiating eight 1-bit Full Adders. Figure 5 show sub blocks used inside the design. For adding three and four values additional circuitry was made by using Full Adders and Half adders. Simple registers were replaced with parallel load registers. First, all the loads were enabled. After the use of each register its load was disabled, so that the value remains saved. The traditional multiplication is done using the Array Multiplier. A 4 bit Array Multiplier was used to get an 8-bit output. This kind of multiplier is selected based on performance after comparison of different multiplier design as shown in table 2.

Table 2 Simulation Power, area, Components for 4 bit different multiplier

	Power		Number of component	Area mm^2	Delay (ns)	Power-delay product
	VDD=5V	VDD=3.3V				
Array Multiplier	4.447	1.85	490	0.533	14.4	64.03
Booth multiplier	11.42	4.72	1278	0.134	16.8	191.8

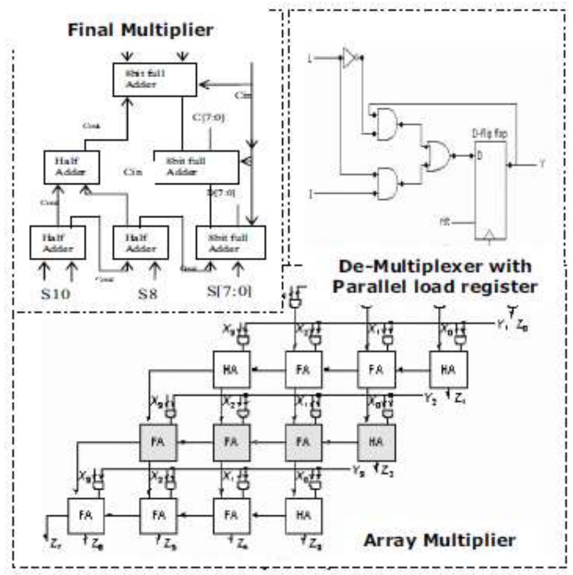
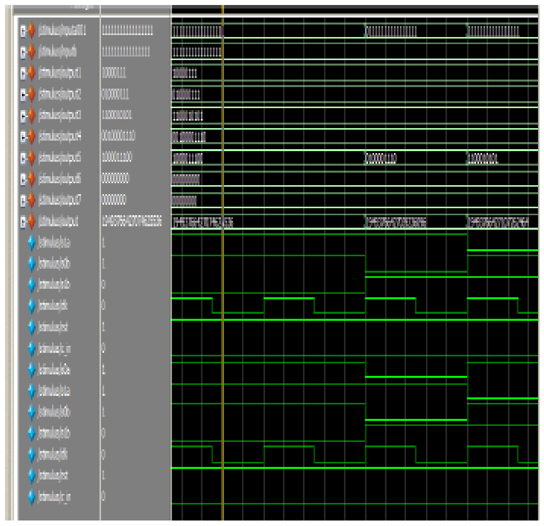


Figure 5 sub-blocks

Verification is completed using the Modelsim simulator. The IO blocks and data format conversion were designed first and tested in the FPGA. The functionality of some of the blocks was verified by simulations before being tested in
Hardware. We used these two numbers:
A = 15 15 15 15 B = 15 15 15 15
The output of the Modelsim simulation to verify functionality is shown in figure 6 and figure 7 although, we have 2 numbers with 4 decimal points the output will be a number with 8 decimal points. The output ranges are from 0 to 7. The top level schematic is shown in figure.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852



Conclusion

In this paper, we presented an optimized implementation of discrete linear convolution. This particular model has the advantage of being fine tuned for signal processing; in this case it uses the mean squared error measurement and objective measures of enhancement to achieve a more effective signal processing model. This implementation has the advantage of being optimized based on operation, power and area. To accurately analyze our proposed system, we have coded our design using the Verilog hardware description language and have synthesized it for FPGA products using ISE, Modelsim and DC compiler for other processor usage. Second, we implemented an illustrative example 4X4 convolver. Similarly, the presented concept can be extended on an NXN case. The functionality of the convolver was tested and verified successfully on a XILINIX SE FPGA and design compiler. The proposed circuit uses only 5mw and saves almost 35% area and it takes 20ns to complete. This shows improvement of more than 50% less power.

References

[1] *John W. Pierre, "A Novel Method for Calculating the Convolution Sum of Two Finite Length Sequences", IEEE transaction on education, VOL. 39, NO. 1, 1996.*

[2] *W. W. Smith, J. M. Smith, "Handbook f Real-Time Fast Fourier Transforms", IEEE Press, 1995, p. 28.*

[3] *R. G. Shoup, "Parameterized convolution filtering in a field programmable gate array," in selected papers from the Oxford 1993 international workshop on field programmable logic and applications on More FPGAs. Oxford, United Kingdom: Abingdon EE&CS Books, 1994, pp. 274–280.*

[4] *Iván Rodríguez, "Parallel Cyclic Convolution Based on Recursive Formulations of Block Pseudocirculant MatricesMarvi Teixeira", IEEE, transaction on signal processing,2008*

[5] *Waldemark J., Becanovic J., Lindblat T., Lindsey C.S.: Hybrid Neural Networks for Automatic Target Recognition, IEEE int. conf. on System, Man and Cybernetics SMC97, Vol. 4, 1997.*

[6] *Miček J.: Unconventional Method of Linear Interpolation, Journal of Information, Control and Management Systems, 2/2005, ISSN 1336-1716.*