Vol 05, Article 11504, December 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

VLSI IMPLEMENTATION OF A FLEXIBLE AND SYNTHESIZABLE FFT PROCESSOR

*RAMAKOTIREDDY THUGUTLA, *Y V BHASKAR REDDY, *K. VEERASWAMY, **A. M. PRASAD
*Department of ECE, QIS College of Engg. & Tech, Ongole
**Department of ECE, JNTUK College of Engg. & Tech, Kakinada
ramakotireddy470@gmail.com, yvbreddy06@yahoo.com, kilarivs@yahoo.com, a_malli65@yahoo.com

ABSTRACT

In this paper, we propose a portable hardware design that implements a Fast Fourier Transform oriented to its reusability as a core. The design has parameterized by the number of samples and the number of data bits. The module has been developed using a radix-2, 4, and 8 decimation in time algorithm of n-point samples. The dimensionless Fast Fourier transform algorithms will work independent of dimensions. The dimensionless Fast Fourier transform algorithms can be configured to compute different dimensional DFTs simply by relabeling the input data and by changing the values of the twiddle factors occurring in the butterfly operations. This observation allows us to design an FFT processor, with minor reconfiguration, which can compute one, two, and three dimensional DFTs. In this paper we design a family of FFT processors, parameterized by the number of points, dimension, number of processors, internal dataflow, and to map different dimensionless FFTs onto this hardware design. Different dimensionless FFTs have different data flows and consequently lead to different performance characteristics. Using a performance model we search for the optimal algorithm for the family of processors that we considered. The resulting algorithm and corresponding hardware design was implemented using FPGA. This project will be designed using Verilog HDL language, simulated using Modelsim Tool and Synthesized using Xilinx Tool.

Index Terms–FFT, very-large-scale integration (VLSI), Xilinx

INTRODUCTION

The FFT procedure for synthesizing and analyzing the Fourier series was given by Cooley and Tukey. It is computationally efficient way to calculate DFT. The wide usage of DFT's in Digital Signal Processing applications is the motivation to implement FFT's. In many applications, the Fast Fourier Transform (FFT) presents an intensive computational task due to the amount of data to be processed. The amount of data depends on the number of points and the dimension of the transform. The engineers and scientists rely on approach techniques such as highly-tuned code for uniprocessors, DSP processors, ASIC, IP cores, and reconfigurable architectures, to meet the performance requirements with respect to other design constraints such as physical space. A list of references to these approaches is provided in [1]. In this study, we focus on mathematical properties of the FFT to help in design of a high-performance hardware. First we base our processor on the dimensionless FFT [2] which allows a single hardware design to compute one, two, and three dimensional DFTS. Second, we provide a framework for systematical mapping alternative FFT algorithms onto the parameterized hardware designs. This is obtained by mapping a mathematical description of the FFT, based on matrix factorizations [3], to hardware that implements flow control and generation of the necessary roots of unity (twiddle factors).

Finally, there are many different FFT algorithms, each with different dataflow, and consequently different performance characteristics. Thus the hardware design becomes on optimization problem over the space of possible FFT data flows. We thus propose a universal FFT engine that is parameterized in terms of the number of points and dimension of the transform, and the choice of the algorithm. We consider a distributed architecture comprised of processing units (with local memory) connected via an interconnection network. We derived a class of optimal FFT dataflow diagrams based on memory-access cost function. The derivation followed a design flow that uses a performance model and its simulation to evaluate the choice of algorithm prior to design of the hardware. Future implementation may use the ASIC technology for the floating-point (complex numbers) arithmetical cores and the FPGA technology for the parameterized flow control units.

This paper proposes the design of 32 & 64-points FFT processing block. The work of the project is focused on the design and implementation of FFT [3] for a FPGA kit. This design computes 32-points FFT and all the numbers follow fixed point format of the type Q8.23, signed type input format is used. The direct mathematical derivation method is used for this design in this project the coding is done in Verilog [8] & the FPGA synthesis and logic simulation is done using Xilinx ISE Design suite 14.1

The Discrete Fourier Transform (DFT) [5] plays an important role in the analysis, design and implementation of the discrete-time signal- processing algorithms and systems it is used to convert the samples in time domain to frequency domain. The Fast Fourier Transform (FFT) is simply a fast (computationally efficient) way to

Vol 05, Article 11504, December 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

calculate the Discrete Fourier Transform (DFT). The wide usage of DFT's in Digital Signal Processing applications is the motivation to implement FFT's.

1. IMPLEMENTATION OF FFT PROCESSOR

Radix-2 is the first FFT algorithm and was proposed by Cooley and Tukey in 1965[1]. Radix-2 FFT algorithm is called the decimation-in-frequency algorithm, is obtained by using the divide-and-conquer approach. To derive the algorithm, we begin by splitting the DFT formula into two summations, one of which involves the sum over the first N/2 data points and the second sum involves the last N/2 data points. The DFT of a give sequence x[n] can be computed using the formula

$$X(k) = \sum_{n=0}^{N-1} x(n) W^{nk} \qquad k = 0,1,2,....N-1$$

The FFT is an efficient algorithm to compute the DFT and its inverse (Cooley and Tukey, 1965; Brigham, 1998). It generally falls into two classes: Decimation In Time (DIT), and Decimation In Frequency (DIF). The DIT algorithm first rearranges the input elements in bit reversed order and then builds the output transform. The DIF algorithm first transforms and then rearranges the output values. The basic idea of these algorithms is to break up an N–point DFT transform into successive smaller and smaller transforms known as a butterfly (basic computational element). The smallest transform used is a 2-point DFT known as radix-2, it process group of 2 samples. The combination of two stages of radix-2 in one stage constitute radix-4 algorithms, it processes group of 4 samples. There are also other decomposition schemes known as split-radix algorithms (Duhamel and Hallmann, 1984). The calculations implied in the basic computational element (butterfly) for radix-2 in DIT algorithm will be given below. A and B are two complex numbers represented as:

A = U + jV B = X + jY Where U and X are real parts and V and Y are imaginary parts.

The FFT is a complicated algorithm, and its details are usually left to those that specialize in such things. This section describes the general operation of the FFT, but skirts a key issue: the use of complex numbers. If you have a background in complex mathematics, you can read between the lines to understand the true nature of the algorithm. Few scientists and engineers that use the FFT could write the program from scratch. In complex notation, the time and frequency domains each contain one signal made up of N complex points. Each of these complex points is composed of two numbers, the real part and the imaginary part. For example, when we talk about complex sample, it refers to the combination of Real and Imaginary. In order to match up when added, the two time domain signals are diluted with zeros in a slightly different way. In one signal, the odd points are zero, while in the other signal, the even points are zero. In other words, one of the time domain signals is shifted to the right by one sample. This time domain shift corresponds to multiplying the spectrum by a sinusoid.

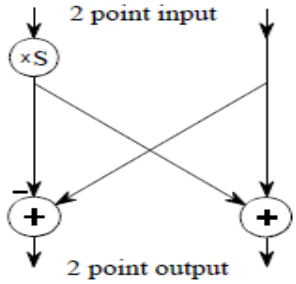

Figure 1: The FFT butterfly

This simple flow diagram is called a **butterfly**. The butterfly is the basic computational element of the FFT, transforming two complex points into two other complex points. Figure 1 shows the structure of the entire FFT. The time domain decomposition is accomplished with a bit reversal sorting algorithm. The frequency domain synthesis requires three loops. The algorithm used in the implementation of VLSI is called FFT radix-2 & 4 and decimation in time. [1] The term decimation in time indicates that the samples are ordered before the FFT calculation. The sort algorithm follows the bit reversal order of the memory location of input samples. The bit reversal order of "110" memory location is "011". Radix-2 & 4 indicates that the DFT was decomposed in two operations that are performed parallel, the FFT butterflies are made using two samples at a time. Decimation in time and radix-2 are terms resulting from the mathematical manipulations that DFT transform FFT, first proposed in [3].The processor uses fixed point FFT as numerical representation [5]. The number of bits reserved for the integer part and the decimal part are the parameters passed before compilation.

Input signals and output

- CLK: It is the input pin that serves as a synchronizer, or time base for the CPU.
- RST: is an input pin that resets the operation any time. When this pin is at high level (logic 1) Registers assume their initial values and state machines that make components go to their initial states.

Vol 05, Article 11504, December 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

- START: an input pin is used to signal the processor when you start your operation.
- RE_DATA and IM_DATA: is a set of pins entry representing the real and imaginary part of the input data. The corresponding number of pins is the number of bits of data and may be changed before compilation.
- RE_OUT and IM_OUT: is a set of pins output where the real and imaginary part of the data arising is transferred. The memory data internal processor are transferred for every cycle clock.

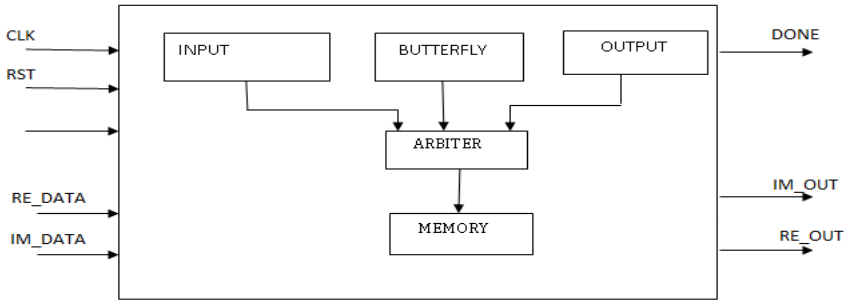

Figure 2. Internal organization of the FFT processor

2. Internal blocks

These blocks were designed as machines states, since this technique allows good predictability of issues encountered in hardware design [7].

- INPUT: is responsible for storing input data in to internal memory. At the beginning of ARBITER operation block signals that block start its operation. It receives data and generates correct address to store in memory.
- BUTTERFLY: This block is the processor core, it is able to convert any size.
- OUTPUT: after the execution of FFT algorithm block initiates the data transfer result, in a given clock cycle.
- MEMORY: is composed of 2 block storage, one for the real part and one for the imaginary part of the data.
- ARBITER: Only one block should be operation time. Thus, this block is responsible for managing the other blocks with signs Start by receiving their signals end and doing interfacing with the memory.

2. FFT ARCHITECTURE

The implementation using FFT with radix-2 decimation and frequency proposed in this Article results in data flow shown in Fig 4, it is possible to observe a FFT of 16 points. The decimation in frequency is also shown through the completion of the operations in the input data by following the reverse order of the bits. Basic representation of the radix-4 FFT butterfly. The implementation using FFT radix-2 decimation-in time proposed in this paper results in the data stream in which it is possible to observe an FFT eight points, and the arrows indicate the sums and terms along arrows are multiplications. W_x^y are exponential complex in the form, called twiddle factor.

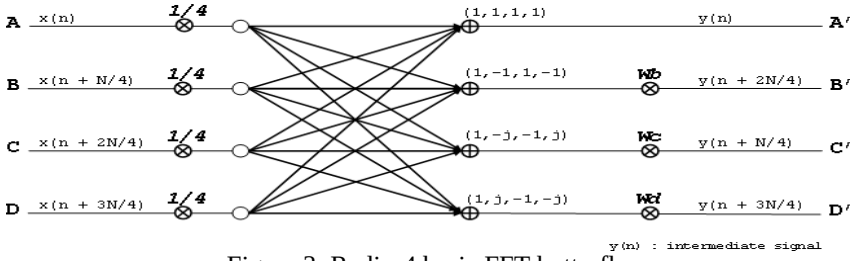

Figure 3: Radix-4 basic FFT butterfly

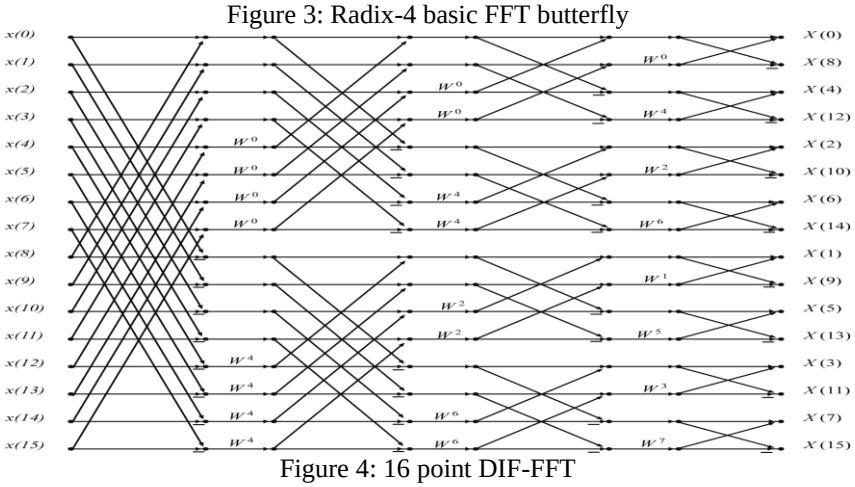

Figure 4: 16 point DIF-FFT

Vol 05, Article 11504, December 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

3. SYNTHESIS REPORT

The FFT processor was synthesized using Xilinx ISE. The results are shown for the calculation of FFT. We used the set of twiddle factors specific to 256 points. This Happens Because the Block Butterfly Continues the Even, the difference is that it uses more bits to control the operations, and there will be more stages and iterations.

Simulation results:

The RTL view of the butterfly structure obtained after the simulation of the 64- point FFT block.

Table 1: Design summary Used In 32-Point FFT

Device Utilization Summary (Estimated Values)			
Logic Utilization	**Used**	**Available**	**Utilization (%)**
Number of Slice Registers	198	607200	0
Number of Slice LUTs	2144	303600	0
Number of fully used LUT-FF pairs	0	2342	0
Number of bonded IOBs	673	700	96
Number of BUFG/BUFGCTRLs	1	32	3
Number of DSP48E1s	36	2800	1

Table 2: Device Utilization of 64 FFT

Device Utilization Summary				
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Notes**
Number of Slice Registers	750	607200	1%	
Number of Slice LUTs	1464	303600	1%	
Number used as logic	1464	303600	1%	
Number used as ROM	0			
Number used as memory	0	130800	0%	
Number of occupied cells	525	75900	1%	
Number of LUT Flip Flop Pairs used	1662			

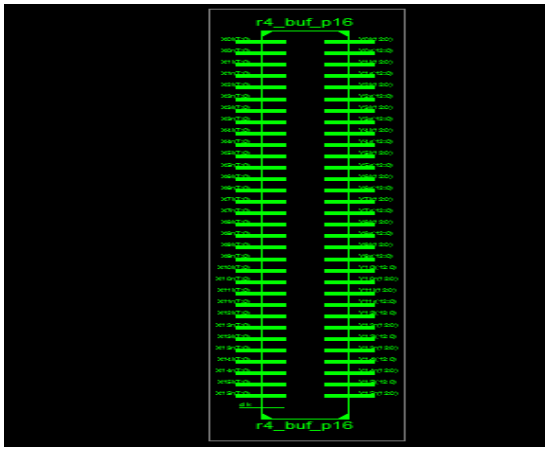

Figure 5: RTL View of a Butterfly Component Used In 32-Point FFT

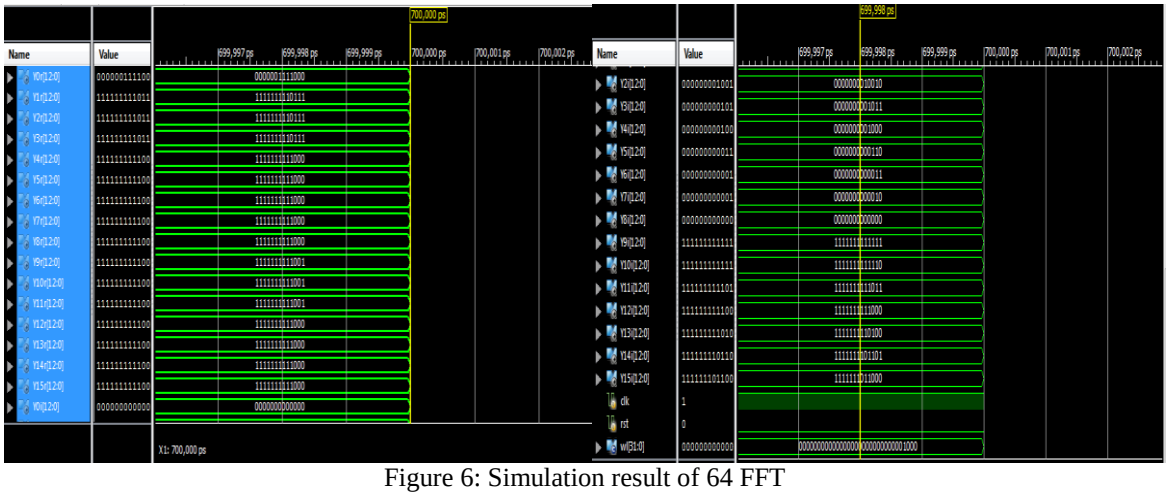

Figure 6: Simulation result of 64 FFT

Vol 05, Article 11504, December 2014
http://ijves.com
International Journal of VLSI and Embedded Systems-IJVES
ISSN: 2249 – 6556

CONCLUSION

This Article presents the implementation, the results of synthesis and details regarding the internal operations of a processor of FFT. This based on FPGA using Verilog HDL as hardware description language and XILINX design and synthesis tool. The dedicated parallel-pipelined FFT processor architecture can process input data at high speed, and the whole system performance can be greatly improved due to adopting a novel simple address mapping scheme. Improvements in this implementation are done as to reduce the execution time and to achieve high throughput for this we have proposed the radix-4 FFT algorithm with DIT.

REFERENCES

[1].Alan V. Oppenheim, Ronald W. And John R. Schafer Buck, Discrete-Time Signal Processing, Prentice Hal, second edition, 1999.
[2].Mandeep Singh Balwinder Singh Pawan Verma, Harpreet Kaur, "VHDL Implementation of FFT / IFFT Blocks for OFDM," International Conference on Advances in Recent Technologies in Communication and Computing, 2009.
[3].James W. Cooley and John W. Tukey, "An algorithm for the machine calculation of complex Fourier series ", Math Comput., p. 297-301,1965.
[4].Volnei A. Pedroni, Circuit Design with VHDL, MIT Press, ISBN 0 - 262-16224-5, 2004.
[5].T. Starr, M. Sorbara, J. M. Cioffi and P. J. Silveramn, DSL Advances, Prentice Hall, 2003.
[6].Rd J.; Ordaz-Moreno A.; Vite-Frias, J. A., Romero-Troncoso, "VHDL core for 1024-point radix-4 fft computation ", International Conference on Reconfigurable Computing and FPGAs, 2005, reconfig 2005, p. 4-24, September 2005.
[7].A. E. Álvarez-Marquina Martinez Icaya C. Gonzaléz-Consejero, V. Rodellar and P. Gonzalez-Vilda, "The portable hardware design of the FFT algorithm ", Latin American Applied Research, 2007.
[8].Randy Yates, "Fixed-point arithmetic: An introduction", 2009, http://www.digitalsignallabs.com.
[9].David Bishop, "Fixed point package user's guide".
[10]. Frank Vahid, Digital Design with RTL Design, VHDL and Verilog, John Wiley and Sons, second edition, 2011.
[11]. Altera, "Altera Corporation", 2011, http://www.altera.com.

AUTHOR BIOGRAPHY

Y.V.Bhasakara reddy is presently working as research scholar in JNTUH Hyderabad, India. He received B.Tech from S.V.University,Tirupathi, India and M.Tech from Bharath University, Chenni, India. He has twelve years of experience of teaching undergraduate students and post graduate students. His research interest is in the area of Microwave Antennas.

Dr.A.M.Prasd is currently working as professor in ECE Department of UCEK Kakinada, India. He received Ph.D from JNTU Hyderbad. He has seventeen years experience of teaching undergraduate students and post graduate students. His research interests are in the areas of Antennas, Network theory and Instrumentation.

Dr.K.Veera Swamy is currently working as professor of ECE and principal of QIS College of Engneering & Technology, Ongole, India. He has fifteen years of experience in teaching undergraduate students and post graduate students. His research interests are in the areas of Image Compression, Image Watermarking, Microwave Antennas, and Networking Protocols.

Ramakotireddy Thugutla was born on 4th Feb 1989 at Chimata, India. He received his, B.Tech in Electronics & Communication Engineering from Priyadarshini College of Engg. & Tech, Nellore, affiliated to JNTU, Anantapur, AP, India and Pursuing M.Tech in DECS at QIS College of Engineering and Technology Ongole, affiliated to JNTU, Kakinada, AP, India. His areas of interest are Digital VLSI design and VLSI image processing.