VLSI Implementation of 2048 Point FFT

Zena Vatsa¹, Sumaya²

Department of Electronics and Communication Engineering, Sharda University, Greater Noida, India

Abstract: Orthogonal Frequency Division Multiplexing (OFDM) is a multi carrier modulation technique. OFDM provides high bandwidth efficiency because the carriers are orthogonal to each other and multiple carriers share the data among themselves. This paper focuses on the core processing blocks of an OFDM system, which are the Fast Fourier Transform (FFT) block and the Inverse Fast Fourier Transform (IFFT). The 8 points IFFT / FFT decimation-in-frequency (DIF) with radix-2 algorithm is analyzed to produce a solution that is suitable for FPGA implementation

Keywords: Orthogonal Frequency Division Multiplexing (OFDM), Fast Fourier Transform (FFT), Inverse Fast Fourier Transform (IFFT), Field Programmable Gate Arrays (FPGA), radix-2, modulation, efficiency.

1. Introduction

А digital communication system involves the transmission of information in digital form from one point to another point. Regardless of the form of communication method, the three basic elements in a communication system consist of transmitter, channel and receiver. The source of information is the messages that are to be transmitted to the other end in the receiver. A transmitter can consist of source encoder, channel encoder and modulation. Source encoder employed an efficient representation of the information such that resources can be conserved. A channel encoder may include error detection and correction code. The aim is to increase the redundancy in the data to improve the reliability of transmission. A modulation process convert the base band signal into band pass signal before transmission. During transmission, the signal experiences impairment which attenuates the signals amplitude and distort signals phase. Also, the signals transmitting through a channel also impaired by noise, which is assumed to be Gaussian distributed component. In the receiver end, the reversed order of the steps in the transmitter is performed. Ideally, the same information must be decoded in the receiving end.

2. FFT Algorithms

FFT is mainly classified in two types, DIT (Decimation in Time) and DIF (Decimation in Frequency). General formula to compute N point FFT is,

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}$$
, $k = 0, ..., N-1$
 $W_N = e^{-j2\pi i N}$

For decimation in time algorithm x(n) is decomposed, and for decimation infrequency algorithm X(k) is decomposed. Both algorithms require same number of operations and require bit reversal at some places during the computation.

3. Radix-2 DIF Algorithm:

Assuming N is power of 2, we separate the index set of eq.(2.1) into two sets. Twoproceed with the decomposition, now we separate the value of k into the sets of even values and odd values. Resolving this will led to X(2k) for k odd and X(2k+1) for k even.

Number of complex multiplication = $(N/2)\log_2 N$ Number of complex addition = $N\log_2 N$



4. Architecture

Most of the application in communication requires operation in real time. This highdemand of real time systems made pipelined architecture a subject of research. DIT and DIF are two different ways one can perform FFT. DIF takes input in normal order and gives output in bit reverse order, so from now onwards we will stick to the DIF algorithm. Radix-2 algorithm is preferred for the VLSI implementation.

Architecture for VLSI Implementation

FFT/IFFT is very much symmetrical architecture. Two point butterfly unit is smallestunit of FFT/IFFT. Modules required implementing signal flow graph for radix-2 DIF algorithm are adder, subtractor, complex multiplier. A simplest architecture to implement FFT/IFFT with required modules and storage elements can be thought. Using thiscomponent for repetitive structure will acquire more area for high speed operation. ForVLSI implementation we are always fighting with area, timing and power constraint. When it is required to get high speed processor in small area, only one idea strikes ineveryone's mind is parallel or pipelined architecture.



Figure 2: Two point FFT calculation (Butterfly Structure)

Pipelined Architecture

To understand pipelined architecture one should have clear idea about what is pipelining?Pipelined in general means sequence of functional stages. Each functional stage takesinput and produces the output which needs to be stored in buffer and work as a new input for the next stage. This way data is flushed through the system. The costs are greater latency and complexity due to the need to synchronize the stages in some way so that different inputs do not interfere. Pipeline will work with full efficiency only if it is filled and emptied at same rate. This can be thought as an assembly in a line industry, where output of one unit is supplied to the other and finally generated product is packed in particular way and ready to dispatch. Important thing to remember is when product is being packed other units are also in functioning stage and producing the output. Pipeline may be synchronous or asynchronous. A synchronous pipeline has a master clock andeach stage must complete its work within one cycle. We will use synchronous pipeline structure for implementation. The minimum clock period is thus determined by the slowest stage. Asynchronous pipeline requires handshaking between stages, so that output of one stage is stored in memory till the previous outputs are not used by the next stage. Literature survey is done for different available pipelined architecture for the VLSI implementation of FFT/IFFT.



5. Implementation and Results

The FPGA implementation is performed using Very High Speed Integrated Circuit (VHSIC) Hardware Descriptive Language (VHDL). This performance of the coding is analyzed from the result of timing simulation using Altera Max Plus II.



Figure 4: RTL schematic of 2 point FFT

Volume 4 Issue 5, May 2015 <u>www.ijsr.net</u>

Xilinx - ISE - C:\Xilinx91i\bin\nt64\bhanu\bhanu.ise - [Simulation	n]										0 X
😧 File Edit View Project Source Process TestBench Simulation Window Help											
□ 🖻 🕫 🕼 🐇 ╘ ╘ 🗙 ぬ 🔎 🔎 🗩 X X 🔎 図 🔊 着 8 🛛 🗗 🖓 🕅 🗞 💽 🖬 🖬 🐨 💭 🖗 🐨 💭 🕼 🕷											
() []] 重] / 3/ 3/ 3/ ④ 20 [④ ④ ☆☆ ‡ 高 (1 □ ()) ズ 1000 ↓ ∞ ↓											
Sources X	Name		1	31.9							
Sources for: Behavioral Simulation	Now:	0		200		400	60	0	80	0	1000
- O bhanu	1000 ns	ľ				j.	ı Î				1000
E [] xc3s200-4tt256	õl a	1									~
the state of	🌏 b	1									
(SII c	1									
💵 Sources 🤭 Snapshots 👔 Libraries											
Processes X											
Hierarchy of and tbw:											
F and_tbw - and_tbw - testbench_arch											
<			4								T.
Processes Sim Hierarchy - and tbw	B and nate vhd	ST Deeinn	Summany 🗔	Simulation							
📙 ang jase waa 👔 balaa ka 👔 balaa ka											
X This is a Lite version of ISE Simulator.											
Simulator is going circuit initialization process.											
s											-
											•
E Console 👩 Errors 🛦 Warnings 📆 Tol Shell 🙀 Find in Files 🔤 Sim Console - and thw											
			-								Time
											time:

Figure 5: Testbench waveform check

6. Conclusion

The Fast Fourier Transform (FFT) and Inverse Fast Fourier Transform (FFT) have been chosen to implement the design instead of the Discrete Fourier Transform and Inverse Discrete Fourier Transform because they offer better speed with less computational time. These methods requires the odd and even samples inputs are process separately before they are combine to give the final output. The result of the computation is in integer bits which might comprises of real and imaginary components. The decimal value of the output if greater than 0.5 is approximated to 1 and vice versa.

References

- Bingrui Wang; Qihui Zhang; TianyongAo; Mingju Huang, "Design of Pipelined FFT Processor Based on FPGA," Computer Modeling and Simulation, 2010. ICCMS '10. Second International Conference on , vol.4, no., pp.432,435, 22-24 Jan. 2010
- [2] Zou Wen; QiuZhongpan; Song Zhijun, "FPGA Implementation of efficient FFT algorithm based on complex sequence," Intelligent Computing and Intelligent Systems (ICIS), 2010 IEEE International Conference on , vol.2, no., pp.614,617, 29-31 Oct. 2010
- [3] Jianing Su; Zhenghao Lu, "Low cost VLSI design of a flexible FFT processor," Solid-State and Integrated Circuit Technology (ICSICT), 2010 10th IEEE International Conference on , vol., no., pp.488,490, 1-4 Nov. 2010
- [4] Yihu Xu; Chung-Hoon Lee; Myong-Seob Lim, "Design of split-radix FFT pruning for OFDM based cognitive radio system," Circuits and Systems

(APCCAS), 2010 IEEE Asia Pacific Conference on , vol., no., pp.524,527, 6-9 Dec. 2010

- [5] GuanwenZhong; Hongbin Zheng; ZhenHuaJin; Dihu Chen; Zhiyong Pang, "1024-point pipeline FFT processor with pointer FIFOs based on FPGA," VLSI and System-on-Chip (VLSI-SoC), 2011 IEEE/IFIP 19th International Conference on , vol., no., pp.122,125, 3-5 Oct. 2011
- [6] Inkeun Cho; Patyk, T.; Guevorkian, D.; Takala, J.; Bhattacharyya, S., "Pipelined FFT for wireless communications supporting 128–2048 / 1536 -point transforms," Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE, vol., no., pp.1242,1245, 3-5 Dec. 2013
- [7] Mangaiyarkarasi, V.; Kumar Charlie Paul, C., "Performance analysis between Radix2, Radix4, Mixed Radix4-2 and Mixed Radix8-2 FFT," Current Trends in Engineering and Technology (ICCTET), 2014 2nd International Conference on , vol., no., pp.430,434, 8-8 July 2014
- [8] Ayhan, T.; Dehaene, W.; Verhelst, M., "A 128:2048/1536 point FFT hardware implementation with output pruning," Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European, vol., no., pp.266,270, 1-5 Sept. 2014
- [9] Zhuo Qian; Nasiri, N.; Segal, O.; Margala, M., "FPGA implementation of low-power split-radix FFT processors," Field Programmable Logic and Applications (FPL), 2014 24th International Conference on , vol., no., pp.1,2, 2-4 Sept. 2014
- [10] Arunachalam, V.; Joseph Raj, A.N., "Efficient VLSI implementation of FFT for orthogonal frequency division multiplexing applications," Circuits, Devices & Systems, IET, vol.8, no.6, pp.526,531, Nov 2014