Design Control Unit & ROM Unit of Low Power FFT Processor for OFDM System

Patel Piyusha B. and Vishal S .Vora

Abstract: This paper specifically addresses the power efficient design of an FFT processor as it relates to OFDM (Orthogonal Frequency Divison Multi Plexing) communication. The FFT/IFFT are widely used in various area such as telecommunication, speech and image processing, medical electronics etc. FFT/IFFT is used as one of the key component in OFDM-based wideband communication systems, like xDSL modems , wireless mobile terminals and remote communication device that rely on limited battery powered operation. Control Unit & ROM Unit of FFT processor is design in VHDL & Simulate the result.

Keywords: **About four key words or phrases in alphabetical order, separated by commas.**

I. INTRODUCTION

Increasing speeds and complexity of wireless communication systems have necessitated the progress and advancement of high performance signal processing elements. Today's emerging technologies require fast processing and efficient use of resources. These resources include power, memory, and chip area. Ongoing research seeks to optimize resource usage as well as performance. Design becomes a balance and compromise of flexibility, performance, complexity, and cost.

This project will specifically address the power-efficient design of an FFT processor as it relates to emerging OFDM communications such as cognitive radio. Cognitive radio is a method of wireless communication by way of dynamically adapting the transmission of multiple subcarriers to changing conditions in the communication channels. These subcarriers are enabled by a modulation scheme known as orthogonal frequency division multiplexing (OFDM). OFDM converts a high data rate signal into multiple lower data rate signals for simultaneous transmission through numerous channels.

The Fast Fourier Transform (FFT) processor is the heart of OFDM that enables its fast and efficient modulation of signals. The FFT algorithm is a fast computation of the Discrete Fourier Transform (DFT) which is an essential component of the modulation scheme used in OFDM. As the FFT processor is the most computationally intensive component in OFDM communication, an improvement in the power efficiency of this component can have great impacts on the overall system. These impacts are significant considering the number of mobile and remote communication devices that rely on limited battery-powered operation. This project will serve as an exploration of current FFT processor algorithms and architectures as well as optimization techniques that aim to reduce the power consumption of these devices.

A. OFDM (ORTHOGONAL FREQUENCY DIVISION MULTIPLEXING)

Orthogonal frequency-division multiplexing (OFDM) is a method of encoding digital data on multiple carrier frequencies. OFDM has developed into a popular scheme for wideband digital communication, whether wireless or over copper wires, used in applications such as digital television and audio broadcasting, DSL broadband internet access, wireless networks, and 4G mobile communications.

OFDM technology is already in use for many communication systems such as ADSL, wireless local area network (WLAN), and multimedia communication services. OFDM is the approach taken to utilize available spectrum. The basic principle of OFDM is to split a high rate data stream into a number of lower rate streams and transmit them simultaneously over a number of subcarriers. that the number of subcarriers could be reduced and that not all used subcarriers will contain useful data, it is again evident that an FFT processor for this application should have the flexibility to accommodate various input lengths. Sub-carrier spacing and symbol duration are selected to obtain orthogonality between subcarriers.

Among these, 1440 and 240 subcarriers are used for data and pilot transmission, respectively." The number of subcarriers could also be based on bandwidth allocated to each user. Seeing that the number of subcarriers could be reduced and that not all used subcarriers will contain useful data, it is again evident that an FFT processor for this application should have the flexibility to accommodate various input lengths. Sub-carrier spacing and symbol duration are selected to obtain orthogonality between subcarriers. It is this orthogonality that has eliminated cross-talk between the sub-channels and the requirement for inter-carrier guard bands. This has also simplified the design of both the transmitter and the receiver.

Other advantages of OFDM as a modulation scheme are listed below:

1) OFDM Advantages:

- Robust against narrow-band co-channel interference
- Robust against inter-symbol interference

Patel Piyusha B. is with Department of ECE, Gujarat Technical University, Ahmadabad, India and *Vishal S .Vora is with* Dept. of ECE, Assistant Professor of Atmiya Institute of Technology & Science, Rajkot, Gujrat Emails: patelpiyusha@yahoo.co.in, vishal.s.vora@gmail.com

(ISI) and fading caused by multipath propagation
- High spectral efficiency
- Efficient implementation using FFT
- Low sensitivity to time synchronization errors
- Tuned sub-channel receiver filters are not required
- Approach Shannon capacity [the theoretical limit of data transmission through an additive white Gaussian noise (AWGN) channel]
- Adaptive resource allocation
- High data rates
- Robustness to multipath delay spread
- Integrates with spectrum sensing

2) OFDM Disadvantages:

- Sensitive to Doppler shift
- Sensitive to frequency synchronization problems
- High peak-to-average-power ratio (PAPR), requiring linear transmitter circuitry, which suffers from poor power efficiency

1.2 OFDM Application:

- Many Communication system such as ADSL (Asymmetric digital subscriber line service used in home networking)
- Broadband Communication Technology used for connecting to the internet
- Wireless local area network(WLAN)
- Multimedia Communication

3) OFDM System:

OFDM Transmitter:

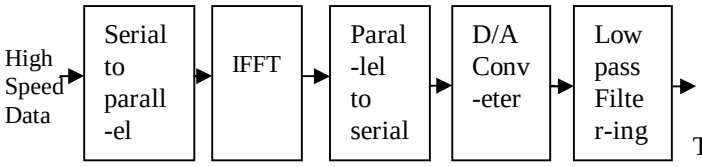


Figure 1.OFDM Transmitter

OFDM Receiver:

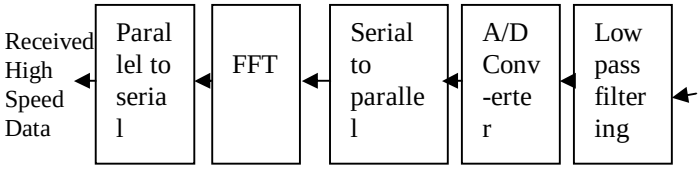


Figure 2.OFDM Receiver

A traditional OFDM system is comprised of several components including a convolutional coder, signal mapper (QPSK, QAM, etc.), FFT processor, parallel-serial converters, and digital-analog converters. A generic block diagram of an OFDM system is shown below. For the purpose of this paper, the discussion of these components will be limited to the FFT processor. In OFDM, a Fast Fourier Transform (FFT) is used to realize the multi-carrier modulation, which reduces the complexity of OFDM systems. With increasing OFDM system speeds and data throughput demands, the implementation of a high speed FFT processor has become the bottleneck of advancement of OFDM techniques.

II. FAST FOURIER TRANSFORM (FFT)

A. FFT Algorithms

The power of the Fourier Transform comes from the fact that even aperiodic signals can be expressed or approximated as a sum of periodic signals. This is important because most of the signals that are of interest in analyzing are not purely periodic in nature. For example, sound recordings of voice and music or transmission of data contain information of varying periods or frequencies. Being able to represent these signals as a sum of periodic signals becomes important when considering the systems through which one would like to pass these signals, specifically, linear time-invariant (LTI) systems. LTI systems are used for much of the signal processing that occurs today. There exist eigen functions for LTI systems where the output of the system is exactly a scaled version of the unmodified input. Sinusoidal periodic signals that can be written in the form e^{jwt} are eigen functions for all LTI systems. Using the Fourier Transform, aperiodic signals that are common in real life systems can be analyzed and processed as periodic eigenfunctions using common LTI systems.

Some of the mathematics involved can be understood in context of a signal, such as sound. A sound signal with a given frequency w can be written mathematically as e^{jwt} . The total sound experienced at the ear could be expressed as the sum of all frequencies each with -magnidute X(w) as

$$x(t)=1/2\pi \int_{-\infty}^{\infty} X(w)\, e^{jwt}\, dw$$

The Fourier Transform giving us the magnitudes of each frequency contained in the signal can be written as

$$X(w)= \int_{-\infty}^{\infty} x(t)\, e^{-jwt}\, dt$$

The Discrete Fourier transform can then be written as

$$X(k)= \sum_{n=0}^{N-1} x(n)\, e^{-j2\pi kn/N},\ 0\le k \le N-1$$

$$X(k)= \sum_{n=0}^{N-1} x(n)\, W_N\ ,\ 0\le k \le N-1$$

Fast Fourier Transform algorithms are mathematical simplifications of the Discrete Fourier Transform (DFT). They exploit symmetries and periodicity in the transform in order to reduce the number of mathematical computations. This is done by a divide and conquer method first invented by Carl Friedrich Gauss around 1805. It was later rediscovered and published by J.W. Cooley and John Tukey and has

become known as the Cooley-Tukey algorithm. This method effectively reduces the computational complexity of the DFT from order $0(N^2)$ to $0(N \log_2(N))$. There have since been many variations of this algorithm aimed at reducing the complexity of the DFT calculations. These families of fast algorithms for computing the DFT are commonly known as FFT algorithms. There are also FFT algorithms that are approximations that trade accuracy for computation speed, but these methods of approximation are not covered in this report. FFT computations can also be classified as Decimation In Time (DIT) orDecimation In Frequency (DIF) where the required computations are identical, but the order is essentially reversed.

B. Different radix FFT:

1) Fixed-radix FFT:

Fixed radix decompositions are algorithms in which the same decomposition is applied repeatedly to the DFT equation. The most common decompositions are radix-2, radix-4, radix-8 and radix-16. An algorithm of radix-r can reduce the order of computational complexity to $0(N \log_r(N))$. In general, higher-radix algorithms perform faster but require higher complexity hardware when compared with smaller-radix algorithms. They also require the input to be of a length exponentially related to the radix. This can potentially lead to wasted computations if input lengths are much smaller than the next exponential size required by the radix.

The basic unit of a DFT calculation is often referred to as a "butterfly" calculation because of the shape of its flow diagram. A radix-2 simplified butterfly flow diagram is shown below. In the diagram, $W_N^{-j2\pi/N}$ which is commonly refered to as the twiddle or phase factor.

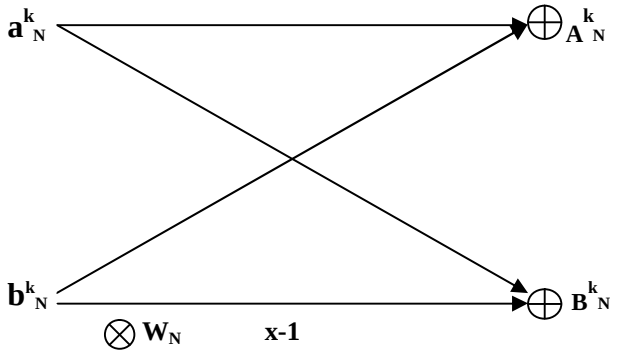


Figure 3. Radix-2 buterfly flow diagram

2) Split-radix FFT:

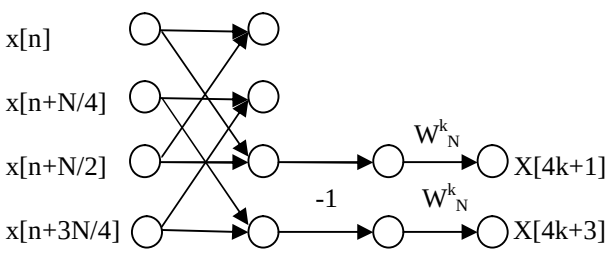


Figure 4. Butterfly signal flow graph of split-radix-2/4 DIF FFT

The split-radix algorithm is a method of blending two or more radix sizes and reordering the sequence of operations in order to reduce the number of computations while maintaining accuracy. "Split-radix FFT algorithms assume two or more parallel radix decompositions in every decomposition stage to fully exploit advantage of different fixed-radix FFT algorithms [2]." As an example of how this reordering takes place, a radix 2/4 flow graph and implementation are shown below. Similar transformations can be done for split-radix-2/8, -2/16, - 2/4/8, and -2/4/8/16 implementations.

3) Mixed –radix FFT:

Mixed-radix refers to using a variety of radices in succession. One application of this method is to calculate FFTs of irregular sizes. For example, a specific application could use radices 2, 3, and 5 to compute an FFT with a size that has those numbers as factors. Mixed-radix can also refer to a computation that uses multiple radices with a common factor. This could be a combination of radices such as 2, 4, and 8. These can be ordered in a way to simplify and optimize calculations of specific sizes or to increase the efficiency of computing FFTs of variable sized inputs.

3. Architecture of FFT

3.1 Simple Architecture:

The choice of hardware architecture also plays a part in optimizing FFT calculations. The figure below shows simple block diagrams of the most commonly used FFT architectures.

Each architecture comes with advantages and disadvantages. Single- and dual-memory architectures have relatively small processor areas but have lower data throughput and require higher clock frequencies. Dual-memory architecture has separate memory for butterfly inputs and butterfly outputs as opposed to one shared memory .A variation of dual-memory involves memory "ping-ponging" where "source and destination memories are interchanged between stages ." This allows data fetching, FFT butterfly computation, and data storing to memory to be combined into a single cycle loop. Parallel architecture can further increase the data throughput allowing for increased processing speed.

Pipeline architecture is the most popular due to its increased throughput and lower clock rates. Compared with memory-based architecture, however, pipeline architecture needs a more complicated control block . The pipeline architecture includes memory components called buffers between calculations stages. These buffers aid in managing data flow and scheduling. An example of a radix-2 signal flow graph is shown above.

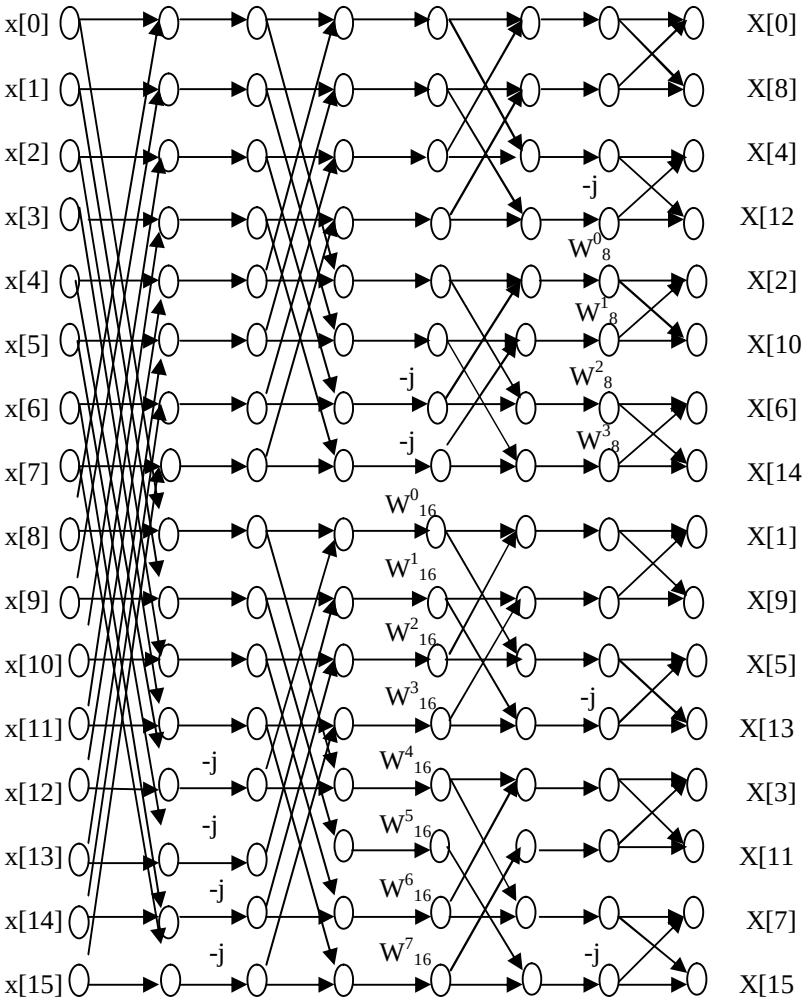


Figure 5. Split-radix-2/4 DIF FFT signal flow graph of a 16-point example

x=input & X= output

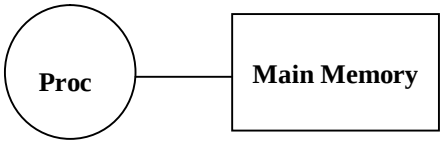


Figure 6 . Single memory architecture block diagram

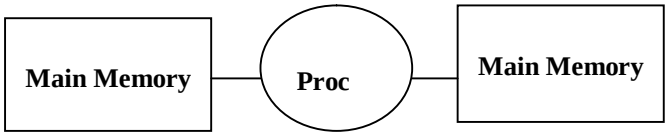


Figure 7 . Dual memory architecture block diagram

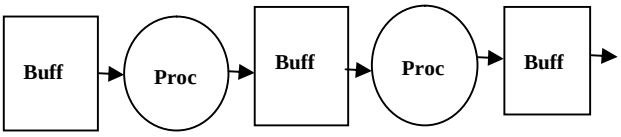
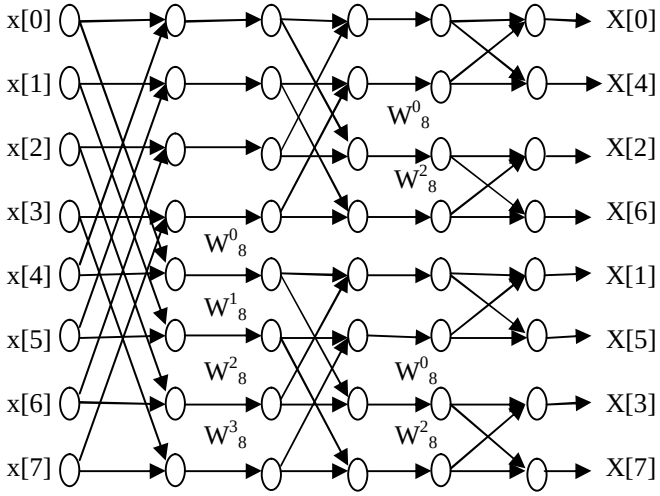


Figure 8. Pipeline architecture block diagram



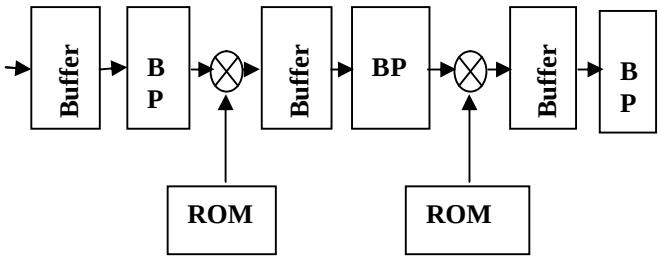


Figure 9 . Projection mapping of the pipeline radix-2 DIF

3.2 FFT processor Block Diagram:

- In the paper we present the on-going research project on design of low-power FFT processors.
- FFT Processor design was to be a 32-bit floating point FFT processor capable of computing a 2048 point FFT.

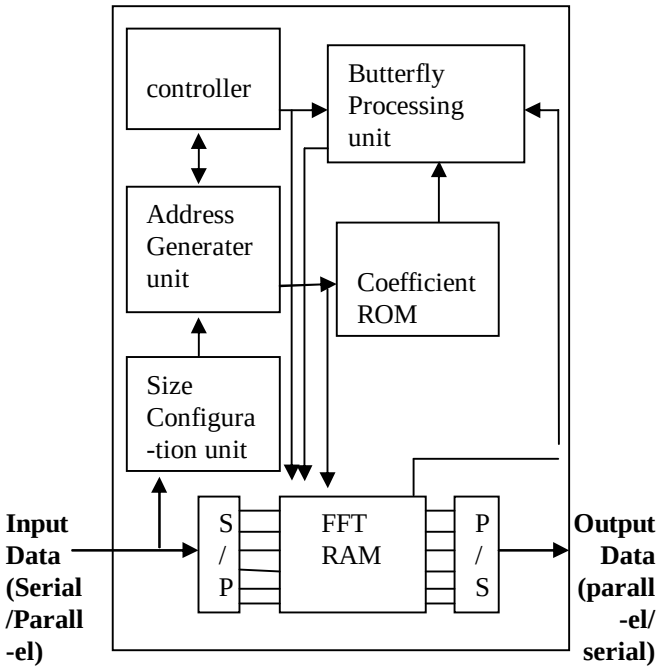


Figure 10. Block diagram of FFT Processor

This baseline was designed following the organization and architecture of an 8-point FFT processor set forth in Specifically, it had a single butterfly processing unit, with supporting ROM, RAM, Counter unit, cycles unit, control unit, and address generation unit

- The baseline processor made use of dual port RAM for the reading and writing of processor inputs and results. Using 32-bit floating point number representation and mathematics, the processor was able to compute the DFT with very little error as can be seen in the data analysis.
- I have run size configuration unit , Butterfly processing unit,ROM Unit & Control Unit in VHDL and simulate the result.

The two primary strategies for pipeline buffering are Multiple-path Delay Commutator (MDC) and Single-path Delay Feedback (SDF). Because MDC architecture requires larger memory size and spends lower hardware utilization, it is usually less desirable than the SDF architecture. SDF memory is series-inseries- out and is comprised of a "first-in-first-out (FIFO) shift register to be the delay element and to feedback the previous input data to compute the DFT ."

In the SFD pipelined architecture, the butterfly element has two operational modes. The first mode is used to fill the shift register with new inputs while transferring previous results to the next stage. The second mode is used to run calculations using the newly acquired inputs and store the subtraction results back into the shift register while the addition is sent to the next stage (see figures below) .

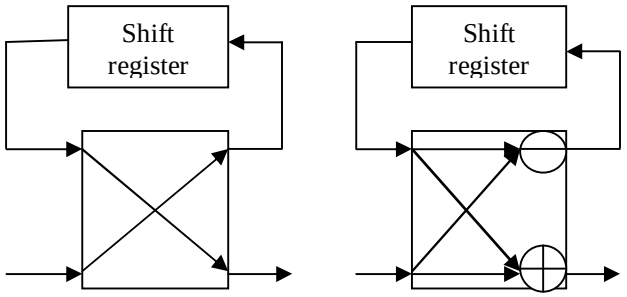


Figure 11. pipelined SDF operational FFT modes

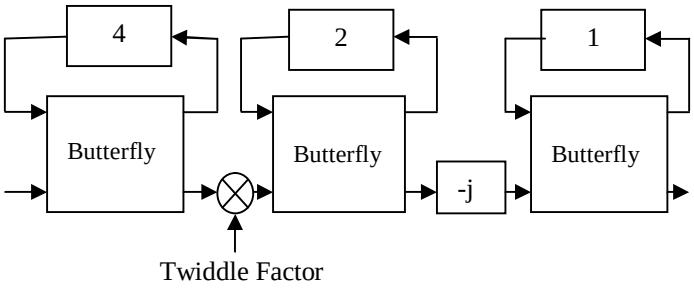


Figure 12. 8-Point radix-2 singal path delay feedback (SDF) architecture

In a fully pipelined architecture, there is one computational block for eachstage in the FFT. For example, in a 2048 point FFT processor, there will be 11 processing units connected together. Inputs and results will progress through the pipelined stages while previous stage processing units are filled with new data. In addition to the computational block, a typical FFT processor includes components such as the address generation block, twiddle factor memory (or generation) block, and control block.

3.3 Platform to be used

VHDL stands for VHSIC Hardware Description Language. VHSIC is an abbreviation for Very High Speed Integrated Circuit, a project sponsered by the US Government and Air Force begun in 1980 to advance techniques for designing VLSI silicon chips. VHDL is an IEEE standard.

Using VHDL for design synthesis:

• Define the design requirement

- Describe the design in VHDL code
- Simulate the source code
- Synthesize, optimize, and fit the design
- Simulate the design
- Implement the design

3.3 Optimization Techniques

Several methods have been employed over the years and in recent times to further optimize the calculation of the DFT. Most endeavors attempt to reduce the number of mathematical operations required to arrive at the solution while others strive to optimize for a resource such as power, speed, area, or memory. Of the many optimization methods previously explored by researchers, a few are summarized below. The following summaries are by no means a comprehensive list of possible modifications but will serve as helpful guidelines in determining desirable implementations. Additional methods exist, and still others have yet to be discovered.

3.4 Power Optimization

There are many applications where small power consumption is desired. Such is the case for devices running on battery for extended periods of time. Power reduction can be achieved by reducing the amount of processing needed or by changing device parameters. If the number of operating logic gates is
lowered, then the amount of power consumed is also reduced. The dynamic power used in CMOS switching can be characterized by the following equation: Pswitch = (1/2)a*C*FC|k*Vdd2 where a is the switching activity factor, Q is the load capacitance, Fcik is the clock frequency, and Vdd is the supply voltage. This equation can provide some insight on the effects of design decisions. For example, parallelization adds capacitance, but it also lowers frequency. You could then lower the supply voltage and realize quadratic power savings. Thus, the pursuit of lower power consumption is assisted by the emergence of more efficient algorithms, yet it can be hindered by increasing clock rates.

3.5 Speed Optimization

There are many ways to optimize the speed of an FFT processor, and this is still an ongoing area of research with constant improvements. In general, the concept for optimizing the speed of an FFT processor is to minimize the number of calculations or steps needed to arrive at the solution. This is done primarily by optimizing algorithms and processor architectures. Part of the difficulty in realizing speed optimization is finding a solution that is efficient at solving a wide variety of problems. Solutions found for a specific application may not be applicable to another of interest. An evaluation must be made to determine the needs of an application and the optimization technique that is most appropriate. Pruning is a method of determining which FFT calculations are unnecessary for a given input. This applies to the case where many input values are zero. Calculations and time can be saved by not using the processor to needlessly multiply and add zeros to other inputs. Conversely, there is a cost of additional computation time to create an index which may or may not be made up for by the FFT computations saved. This method is best suited for inputs with regularly occurring zeros and may not be desired for systems with very unpredictable or changing inputs such as cognitive radio.

As it relates to speed optimization, pipelining splits the calculations into multiple distinct steps and allows calculations on the next inputs to be started while previous inputs are shifted to the following step. This allows for increased data throughput with less memory. The hardware required for pipelining involves more controls/timers, but the calculation elements have more regularity and modularity.

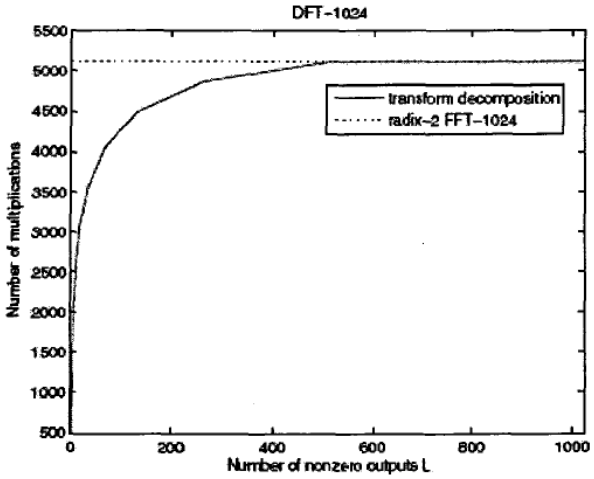


Figure 13. Comparison of computation for radix-2 FFT and transform decomposition for N=1024

This modularity can aid in performing efficient FFT calculations on a variety of input sizes. Pipelining can be implemented on individual processes in the processor as well as on the processor architecture as a whole.When architecture is referred to as being pipelined, it consists of multiple processing units that are staged with inputs and outputs timed in a way that enables the next stage to start calculations on current outputs while the current stage receives new data for calculation. Transform decomposition is a method developed in for analyzing the calculation needs for a given input and determining which computations are unnecessary. Its purpose is similar to pruning, but its implementation is different. It is shown that for inputs with a large number of zero values, the number of multiplications performed is dramatically decreased. This method could be advantageous in OFDM if many of the subcarriers are set to zero because of under-utilized or noisy channels. A graph showing the number of multiplications versus number of non-zero inputs using this technique is shown below. Ideally, cognitive radio will be able to dynamically adjust OFDM subcarriers to prevent this large number of irrelevant inputs.

Parallel-butterfly and dual-butterfly are terms used in to describe methodsof parallel computations in a pipeline that includes a separate butterfly for calculating steps involving a twiddle factor that equals 1. This separate butterfly consists of only additions and can be computed in parallel with only

the small increase of adders to hardware. This results in slightly faster computation with minimal increase to chip size. Another parallel architecture dubbed the "Supper-FFT" is a complicated architecture involving an array of pipelined FFT processors each with local cache but with shared global memory. The goal of this processor was to increase data bandwidth while decreasing power consumption. Part of the power reduction is attributed to the strategic placement of devices on the chip to reduce losses due to long wire lengths in the interconnect network.

Much work has been done to simplify the mathematical algorithms that define the FFT processors. The results of this work have the potential to increase processor speeds, reduce power consumption, reduce chip area, and possibly reduce memory size. So far, no theoretical lower limit to the number of computations needed has ever been determined. The fixed-radix, divide and conquer method, has long been thought of as an efficient method of computing the DFT, but several advancements have increased the efficiency even further.

One method, sometimes referred to as the lifting scheme, can change a radix-4 computation from 4 real multiplies and 2 real adds to 3 real multiplies and 3 real adds. While this doesn't reduce the total number of mathematical operations needed, it does slightly reduce the complexity of computer processor computations leading to a faster, more efficient algorithm. In 1968, the algorithm known as split-radix was first presented and held the record for lowest published count for power-of-two size N calculation, which requires 4A/log2A/ - 6A/ + 8 real multiplications and additions. This record was broken in 2007 by two groups, Johnson/Frigo and Lundy/Van Buskirk. Each was able to reduce the number of real multiplications and additions to approximately 34/9 $N \log_2(N)$. These new algorithms were modified versions of the split radix that previously held the record. Another optimization scheme developed is known as the Rader-Brenner algorithm of 1976 which factors the algorithm to use purely imaginary twiddle factors "reducing multiplications at the cost of increased additions and reduced numerical stability." While each of these is an optimization for the FFT, it is important to remember that a general optimization may not be the best method for a specific application, and a specific optimization may not be the best for a general application. Care must be taken to use the correct optimization for any one application.

3.6 Twiddle Factor Storage/Generation

The storage and/or generation of twiddle factors (FFT coefficients) has been an item of interest to many processor designers because of its impacts on not only speed, but memory and area as well. Traditional methods of dealing with
twiddle factors include either storing or computing the needed values. All needed values can be stored in memory as a look-up table (LUT) or only e1 can be stored while the rest of the values are computed from it. Clearly, one method requires more memory and the other requires more computation. A system for efficiently storing and generating twiddle factors that balances memory usage and computational needs is proposed in. This implementation exploits coefficient symmetries to store 1/8 of the values for radix-8 (1/4 for radix-4) and computes the rest. The values to be stored are strategically chosen to reduce the amount of computations required to generate the rest. This method requires more hardware and introduces a delay, but results are shown to reduce area, memory, and power for FFT computations greater than 1000 points. The equations and block diagrams for the N/4 and N/8 coefficient generators for reference.

III METHODOLOGY

A. Purpose and Goal

With mobile devices increasing in complexity and capability, the power requirements to operate these devices are continually increasing. Given this trend, low power electronics with the capability of extending run time on limited battery capacity are more important than ever. Based on the growing need for mobile devices to have advanced communication capabilities while extending battery life, this project design had a primary focus on power reduction of the FFT processor. Implementation techniques with potential power savings were reviewed and compared when deciding which designs would have considerable impact when targeting OFDM applications.

This project documented the power savings obtained by the design and implementation of many of these techniques. Care was taken to achieve a highly organized processor design with modular components that can be easily modified and analyzed independent from other components. This led to a completely synthesizable solution that could serve to facilitate future research in the area of FPGA design of FFT processors.

B. Design Process and Baseline

The first task was to come up with a design that would serve as the baseline against which to compare all other design choices and implementations. This baseline design was to be a 32-bit floating point FFT processor capable of computing a 2048 point FFT. This baseline was designed following the organization and architecture of an 8-point FFT processor set forth in . Specifically, it had a single butterfly processing unit, with supporting ROM, RAM, Counter unit, cycles unit, control unit, and address generation unit. The baseline processor made use of dual port RAM for the reading and writing of processor inputs and results. This 2048 point FFT processor was successfully implemented and tested for computational accuracy.

C. FFT Processor Components

1. Size Configuration Unit

2. Butterfly Processing Unit

3. ROM Unit

4. Control Unit

5. Address Generator Unit

The address generation unit is where most of the complex control actually takes place. This unit takes on the complex role of determining the correct information that needs to go to each unit at specific times. One process completed is to get the correct inputs from the RAM to the butterfly processing unit during the correct cycles. It also determines which twiddle factor is needed from ROM memory during each of the calculations. The final outputs are then coordinated to be written back to the appropriate locations in memory.

a. Butterfly Counter: Keeps track of how many butterflies have been computed in the current stage
b. Stage Counter: Keeps track of how many stages have been completed
c. IO/Stage Done signal generator: Signals the completion of the current process to the control unit.
d. Address Index Generator: Uses size, stage, and FFT information to determine needed address information
e. 10 Address Generator: Passes unmodified addresses during input operation and bit-reversed addresses during output operation.
f. Address Shifters: Store the input address used for the butterfly processor during the pipelined process so that the same address can be used cycles later to store the butterfly processor results.
g. Multiplexers: Used to select between addresses generated for input/output and those generated for the butterfly processor operation.
h. ROM Address Generator: Determines the address location for the appropriate twiddle factor used in each butterfly calculation.

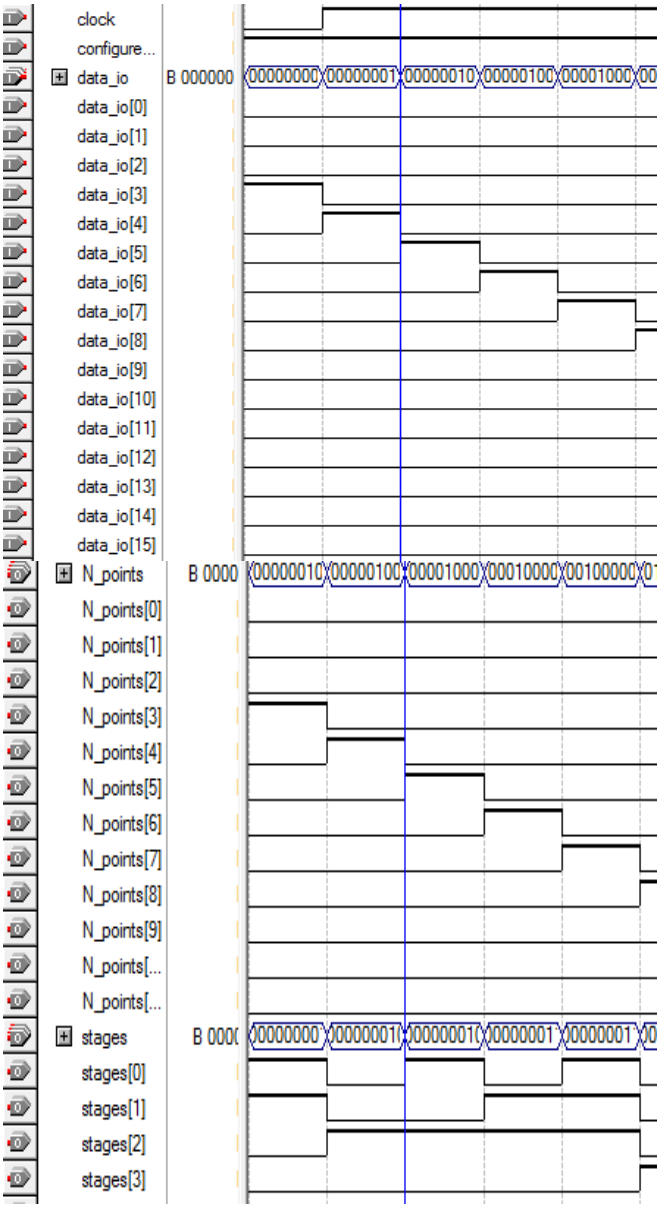
6. RAM Unit

The RAM unit consists of dual-port RAM capable of simultaneous read and write to meet the demands of the pipelined butterfly calculation process. The RAM is used for storing the inputs and results of each stage and has 4096 (2N) available banks of 16 bits each. Real and Imaginary components of the complex data used by the processor are stored in independent memory banks separated by N addresses. In this manner, the first N addresses of the RAM are used to store the real values, while the second N addresses of the RAM contain the imaginary values of the complex inputs and results.

IV. SIMULATION RESULT

1. Size Configuration Unit

The size configuration unit is passed information received in the first word of the incoming signal. This input contains the size of the FFT to be computed. The configuration unit passes this information along with the number of computational stages needed to the other components. All of the components are programmed to automatically decrease or increase calculations or functions performed to match the current input size.

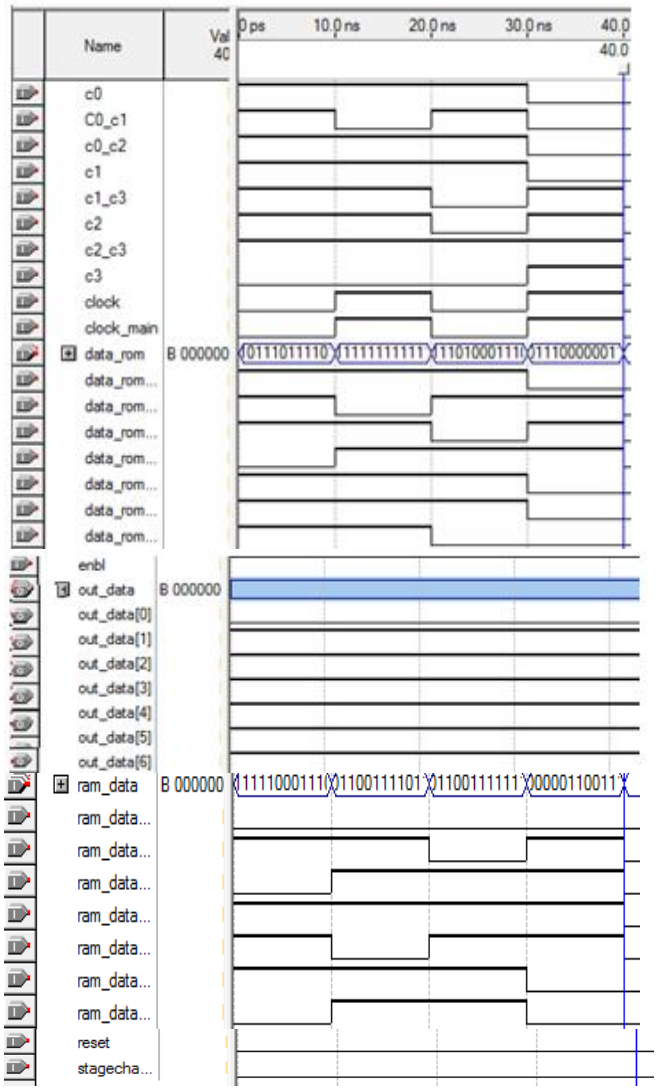


2. Butterfly Processing Unit

The butterfly processing unit computes an inplace radix-2 FFT calculation of two 16-bit fixed point numbers as described previously.

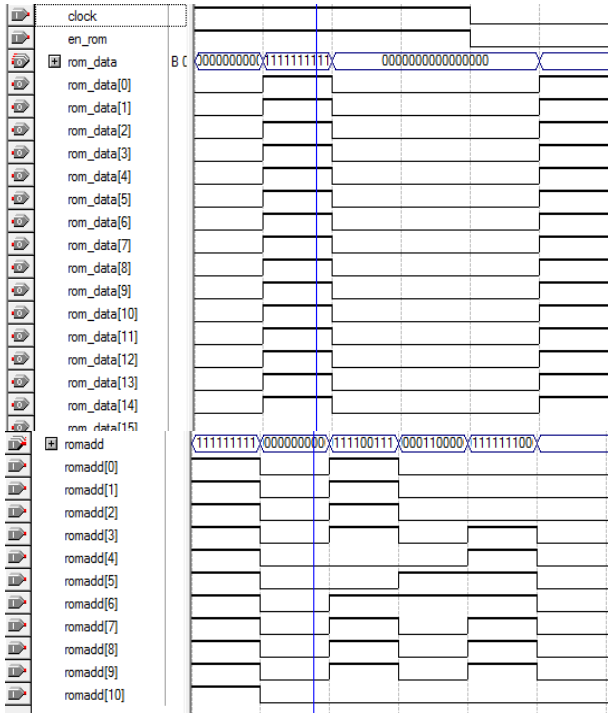
a. Flip Flops: Used to control data flow and break the process into separate pipelined steps.
b. Multiplexer Units: Used as selectors that control which data is passed on during each cycle in the pipeline.
c. Negate Unit: Used to complete necessary mathematical functions of the DFT equation.

d. Multiplier Unit: Used to complete necessary mathematical functions of the DFT equation.
e. Adder Units: Used to complete necessary mathematical functions of the DFT equation.



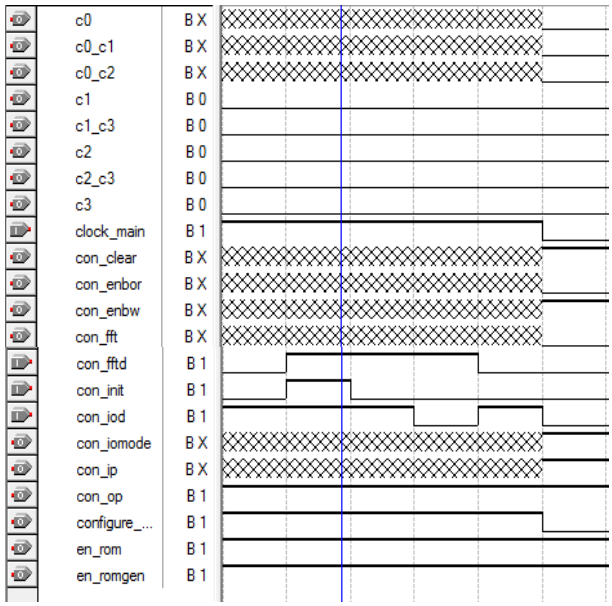
3. ROM Unit

The ROM unit stores N/2 complex twiddle factor values that were pre-computed and converted to fixed point binary representation.

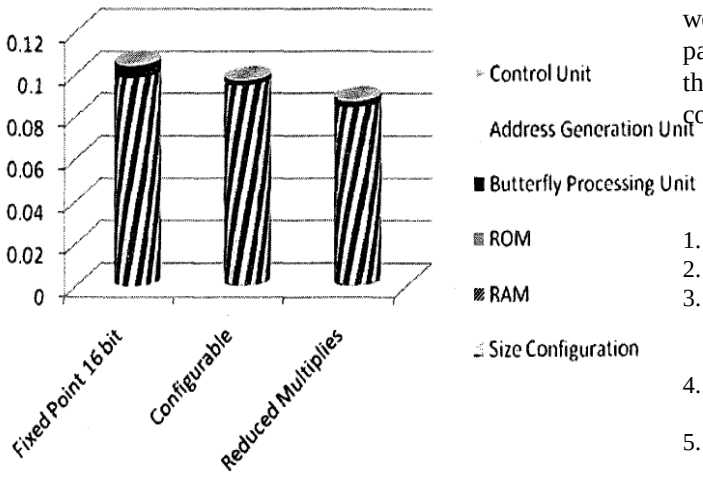


4. Control Unit

The control unit is an eight stage finite state machine that receives input signals from the other components which indicate the completion of tasks performed. Based on these inputs, the controller progresses through its states from configuration to memory access and calculations to final output. Signals are sent to other components indicating the current process.



A. Fixed Point FFT power Comperison



B. FFT Processor power (W) Comperison :

	Floating point 32-bit	Fixed point 16-bit	configuration	Reduced multipli es
Total Quiescent Power	0.10313	0.10313	0.10313	0.10313
Total Dynamic Power	0.48384	0.13037	0.12415	0.11355
Total power	0.58696	0.2335	0.22728	0.21667
Junction temp	25	25	25	25
FFT processor	0.44942	0.10804	0.1047	0.09538
Size configurat io-n unit	0	0	0.00076	0.0009
RAM	0.14729	0.09827	0.09401	0.08366
ROM	0.00022	0.00022	0.00022	0.00022
Butterfly Processin g unit	0.27876	0.00565	0.00234	0.00274
Address Generator unit	0.00031	0.00037	0.00047	0.00083
Control unit	0.00063	0.00083	0.00012	0.0001

V. Conclusion

This paper considers numerous power-saving techniques available for FFT processor design while taking into account the current technology requirements that are emerging for tomorrow's communications. As performance requirements continue to increase while devices are becoming more and more portable, the need for power efficiency becomes ever more important.

Acknowledgement

We have taken efforts in this paper. However, it would not have been possible without the kind support and help of many individuals. we have highly indebted to **Prof. A.M.Kothari , AITS, Rajkot** for his guidance and constant supervision as well as for providing necessary information regarding the paper. we would like to express us gratitude towards him for their kind co-operation and encouragement which help me in completion of this paper.

References

1. www.google.com
2. http://en.wikipedia.org
3. Thomas Lenart and Viktor b a l l "A 2048 COMPLEX POINT FFT PROCESSOR USING A NOVEL DATA SCALING APPROACH"Jan 2000.
4. Hoffman and Altamiro Susin and Luigi Carro "A bit-serial FFT processor" Electroscience engineering.in 2005.
5. S. Y. Lee and C. C. Chen, "VLSI implementation of programmable FFT architectures for OFDM communication system," in IWCMC proceedings,2006
6. S. Reza Talebiyan and Saied Hosseini-Khayat "POWER ESTIMATION OF PIPELINE FFT PROCESSORS"in 2009.
7. DR. D. RAJAVEERAPPA1, K. UMAPATHY2 "Low-Power and High Speed 128-Point Pipline FFT/IFFT Processor for OFDM Applications" Department of ECE at Loyola Institute of Technology,Chennai,India.
8. Y. Zhao, A. T. Erdogan, and T. Arslan, "A novel low-power reconfigurable FFT processor," Institute of Electrical and Electronics Engineers, 2005.[Online] Available: http://ieeexplore.ieee.org [Accessed: Jul. 30, 2008].
9. Q. Zhang, And B.J. Kokkeler and Gerard J.M. Smit, "Adaptive OFDM System Design For Cognitive Radio", Department of Electrical Engineering, Mathematics and Computer Science University
10. Weidong Li and Lars Wanhammar "A Pipeline FFT Processor" Electrical Engineering Dept.Linkoping University
11. Bevan M. Baas "AN APPROACH TO LOW-POWER, HIGH-PERFORMANCE, FAST FOURIER TRANSFORM PROCESSOR DESIGN" February 1999.

Patel Piyusha B. has pursuing M.E. in Electronics & Communication from Atmiya Institute of Technology & Science,Rajkot,Gujrat . Her research interest includes Digital Signal Processing & VLSI

Vishal S. Vora has pursuing Ph.D (Embedded system).he is right now working as an assistant professor in department of Electronics & Communication at Atmiya Institute of Technology & Science, Rajkot, Gujarat . He had published more than 10 papers in reputed national or international journal and conference. He had attended and delivered expert talk in many workshop, STTPS and seminars of embedded system