FPGA Implementation of FFT Algorithm for OFDM Based IEEE 802.16d (Fixed WiMAX) Communications

K. Harikrishna, T. Rama Rao, and Vladimir A. Labay

***Abstract*—The IEEE 802.16d communication standard uses orthogonal frequency division multiplexing (OFDM). In the widely used OFDM systems, the fast Fourier transform (FFT) and inverse fast Fourier transform pairs are used to modulate and demodulate the data constellation on the sub-carriers. In this paper, a high level implementation of a high performance FFT for OFDM modulator and demodulator is presented. The design has been coded in Verilog and targeted into Xilinx Spartan3 field programmable gate arrays. Radix-2^2 algorithm is proposed and used for the OFDM communication system. The design of the FFT is implemented and applied to fixed WiMAX—IEEE 802.16d communication standard. The results are tabulated and the hardware parameters are compared. The proposed architecture is least in number of multipliers used and the memory size, and second to the least in number of adders used.**

***Index Terms*—Fast Fourier transform, orthogonal frequency division multiplexing, radix conversion, Verilog.**

doi: 10.3969/j.issn.1674-862X.2010.03.001

1. Introduction

OFDM technology is used for many communication systems such as asymmetric digital subscriber line (ADSL), wireless local area network (WLAN) or multimedia communication services[1]. One of the key components in OFDM system is the fast Fourier transform (FFT). There are more and more communication systems requiring higher points FFT and higher symbol rates. The requirement establishes challenges for low power and high speed FFT design with large points. In our target application, the IEEE 802.16d (fixed WiMAX) standard requires the OFDM symbol rates from 1.75 MHz to 20 MHz and the FFT up to 2048 points[1]. There are in general two approaches in implementing the FFT for OFDM processing: the pipeline processing and the memory-based recursive processing[1].

The FFT algorithm eliminates the redundant calculation which is needed in computing discrete Fourier transform (DFT) and is thus very suitable for efficient hardware implementation[2]. In addition to computing efficient DFT, the FFT also finds applications in linear filtering, digital spectral analysis and correlation analysis, ultra wide band (UWB) applications, etc. A hardware oriented radix-2^2 algorithm[3] is developed by integrating a twiddle factor decomposition technique in dividing and conquering approach to form a spatially regular signal flow graph (SFG). Mapping the algorithm to the cascading delay feedback structure leads to the proposed architecture[4].

In the search for high performance FFT, this paper presents a pipelined architecture called R2^{2}SDF, of which the implementation on field programmable gate arrays (FPGAs) uses Verilog hardware description language (HDL). The next section describes architecture and design methodology, followed by its implementation in Verilog code and utilization, performance and implementation in OFDM systems. Finally, this paper concludes with a comparison of hardware requirement of R2^{2}SDF and several other popular pipeline architectures.

2. Architecture and Design Methodology

2.1 Radix-2^2 Decimation in Frequency FFT Algorithm

A useful state-of-the-art review of hardware architectures for FFTs was given by He *et al.*[5] and different approaches were put into functional blocks with unified terminology. From the definition of DFT of size N[6],

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

where W_N denotes the primitive Nth root of unity, with its exponent evaluated modulo N, $x(n)$ is the input sequence and $X(k)$ is the DFT. He[5] applied a 3-dimensional linear index map,

$$n=\left\langle\frac{N}{2}n_1+\frac{N}{4}n_2+n_3\right\rangle_N$$
$$k=\left\langle k_1+2k_2+4k_3\right\rangle_N \tag{2}$$

Manuscript presented at 2010 International Conference on Signal Acquisition and Processing (ICSAP 2010), Bangalore, India, February 9-10, 2010; recommended by TPC of ICSAP 2010, revised July 15, 2010.

K. Harikrishna is with Narayana Engineering College, Nellore, AP, India. (e-mail: kamathamhk@yahoo.com).

T. R. Rao is with SRM University, Chennai, TN, India. (e-mail: ramarao@ieee.org).

V. A. Labay is with Gonzaga University, Spokane, WA, USA.

and common factor algorithm (CFA) to derive a set of 4 DFTs of length $N/4$ as

$$X\left(k_1+2k_2+4k\right)_3=\sum_{n_3=0}^{N/4-1}\left[H\left(k_1,k_2,n_3\right)W_N^{n_3(k_1+2k_2)}\right]W_{N/4}^{n_3k_3} \quad (3)$$

where n_1, n_2, n_3 are the index terms of the input sample n and k_1, k_2, k_3 are the index terms of the output sample k and $H(k_1, k_2, n_3)$ is expressed in (4).

$$H(k_1,k_2,n_3)=\left[x(n_3)+(-1)^{k_1}x\left(n_3+\frac{N}{2}\right)\right]+(-j)^{(k_1+2k_2)}\times\left[x\left(n_3+\frac{N}{4}\right)+(-1)^{k_1}x\left(n_3+\frac{3N}{4}\right)\right]. \quad (4)$$

Equation (4) represents the first two stages of butterflies with only trivial multiplications in the signal flow graph (SFG), as Butterfly I (BF2I) and Butterfly II (BF2II). Full multipliers are required after the two butterflies in order to compute the product of the decomposed twiddle factor $W_N^{n_3(k_1+2k_2)}$ in (3). Note the order of the twiddle factors is different from that of radix-4 algorithm.

Applying this CFA procedure recursively to the remaining DFTs of length $N/4$ in (3), the complete radix-2^2 decimation-in-frequency (DIF) FFT algorithm is obtained. The corresponding FFT flow graph for N=16 is shown in Fig. 1, where small diamonds represent trivial multiplication by $W_N^{N/4}=-j$, which involves only real-imaginary swapping and sign inversion[5].

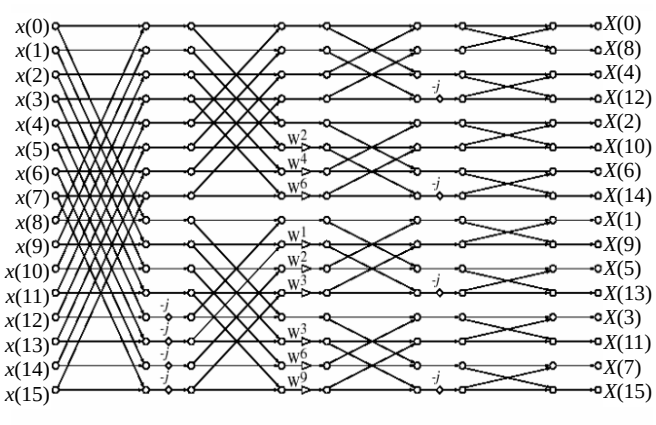


Fig. 1. Radix-2^2 DIF FFT flow graph for N=16.

2.2 Radix-2^2 FFT Architecture

Mapping radix-2^2 DIF FFT algorithm derived to the radix-2 SDF architecture, a new architecture of R2^{2}SDF approach is obtained[3]. Fig. 2 outlines an implementation of the R2^{2}SDF architecture for N=1024, note the similarity of the data-path to R2SDF and the reduced number of multipliers. The implementation uses two types of butterflies: one identical to that in R2SDF, the other contains also the logic to implement the trivial twiddle factor multiplication, as shown in Fig. 3 (a) and (b) respectively[3].

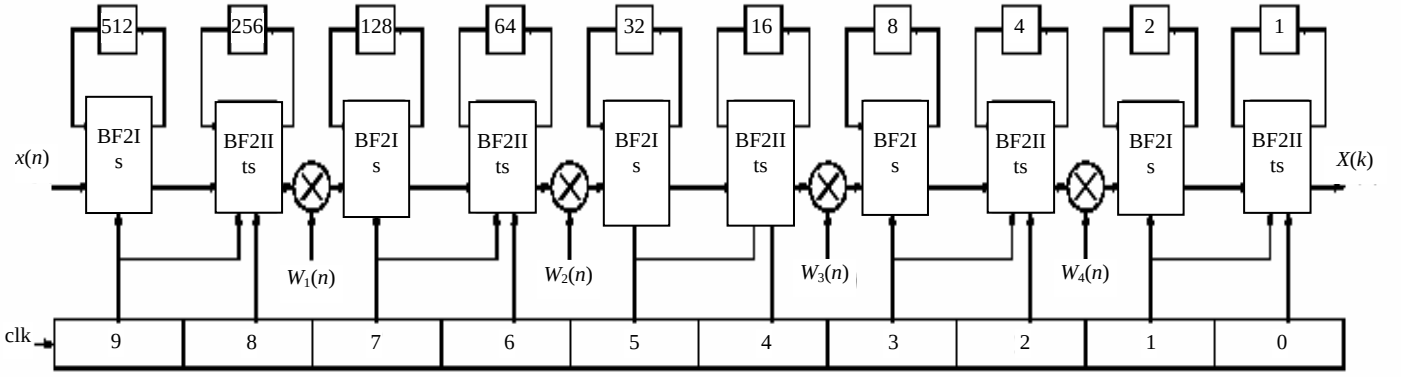
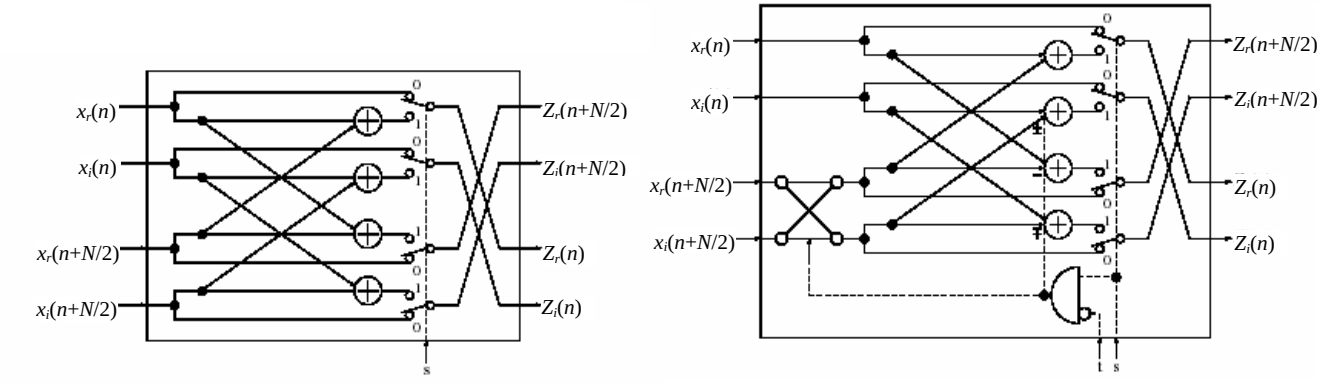


Fig. 2. R2^{2}SDF pipeline FFT architecture for N=1024.



(a) BF2I (b) BF2II

Fig. 3. Butterfly structure for R2^{2}SDF FFT.

Due to the spatial regularity of radix-2^2 algorithm, the synchronization control of the processor is very simple. A ($\log_2 N$)-bit binary counter serves two purposes: synchronization controller and address counter for twiddle factor reading in each stage. With the help of the butterfly structures shown in Fig. 3, the scheduled operation of the R2^2SDF processor in Fig. 2 is as follows.

On the first $N/2$ cycles, the 2-to-1 multiplexers in the first butterfly module switch to position "0", and the butterfly is idle. The input data from left is directed to the shift registers until they are filled. On next $N/2$ cycles, the multiplexers turn to position "1", the butterfly computes a 2-point DFT with incoming data and the data stored in the shift registers.

$$Z_1(n) = x(n) + x\left(n + \frac{N}{2}\right)$$
$$Z_1\left(n + \frac{N}{2}\right) = x(n) - x\left(n + \frac{N}{2}\right), \quad 0 \le n < \frac{N}{2}. \quad (5)$$

The butterfly outputs $Z_1(n)$ and $Z_1(n+N/2)$ are computed according to the equations given in (5). $Z_1(n)$ is sent to apply the twiddle factors, and $Z_1(n+N/2)$ is sent back to the shift registers to be "multiplied" in still next $N/2$ cycles when the first half of the next frame of time sequence is loaded in. The operation of the second butterfly is similar to that of the first one, except the "distance" of butterfly input sequence are just $N/4$ and the trivial twiddle factor multiplication has been implemented by real-imaginary swapping with a commutator and controlled add/subtract operations, as shown in Fig. 3 (a) and (b), which requires two bit control signal from the synchronizing counter. The data then goes through a full complex multiplier working at 75% utility, and accomplishes the result of first level of radix-4 DFT word by word. Further processing repeats this pattern with the distance of the input data decreases by half at each consecutive butterfly stages. After N–1 clock cycles, the result of the complete DFT transform streams out to the right, in bit-reversed order. The next frame of transform can be computed without pausing due to the pipelined processing of each stage.

3. Introduction to Verilog

Verilog HDL was introduced by Gateway Design Automation in 1984 as a proprietary hardware description and simulation language[7]. The introduction of Verilog-based synthesis tools in 1988 by then-fledging Synopsys and the 1989 acquisition of Gateway by Candence Design Systems were important events that led to wide-spread use of the language[7].

Verilog synthesis tools can create logic-circuit structures directly from Verilog behavioral descriptions, and target them to a selected technology for realization. Using Verilog, you can design, simulate, and synthesize anything from a simple combinational circuit to a complete microprocessor system on a chip.

Verilog started out with and still has the following features[7]:

1) Designs may be decomposed hierarchically.
2) Each design element has both a well-defined interface and a precise functional specification.
3) Functional specifications can use either a behavioral algorithm or an actual hardware structure to define initially by an algorithm, to allow design verification of higher level elements that use it; later, the algorithmic definition can be replaced by a preferred hardware structure.
4) Concurrency, timing, and clocking can all be modeled. Verilog handles asynchronous as well as synchronous sequential-circuit synthesis.
5) The logical operation and timing behavior of a design can be simulated.

Thus, Verilog started out as a documentation and modeling language, allowing the behavior of digital-system designs to be precisely specified and simulated. The Verilog language specification allows multiple modules to be stored in a single text file. When one Verilog module instantiates another, the compiler finds the other by searching the current workspace, as well as predefined libraries, for a module with the instantiated name. Thus, when using Verilog-1995, there should be only one definition of each module, usually in a file with the same name as the module.

However, Verilog-2001 actually allows you to define multiple versions of each module, and it provides a separate configuration management facility that allows you to specify which one to use for each different instantiation during a particular compilation or synthesis run. This lets you try out different approaches without throwing away or renaming your other efforts. All these features of Verilog will help better in simulation and synthesis of our proposed architecture.

4. Implementation Using Verilog

The R2^2SDF presented above has been fully coded in Verilog HDL. Once the design is coded in Verilog, the Modelsim XEIII 6.2c compiler and the Xilinx Foundation ISA Environment 9.1i generate a net-list for FPGA configuration. The net-list can then be downloaded into the FPGA using the same Xilinx tools and Texas Instruments prototyping board.

From the architecture of R2^2SDF in Fig. 2, the butterfly blocks BF2I and BF2II are described as building blocks in Verilog code. Booth multiplication algorithm for signed binary numbers is used for complex multipliers. Thus, the overall latency of the real implementation varies as the processing word length changes[3]. Look-up-table (LUT)

based random access memories (RAMs) and flip-flops are used to implement feedback memory of the very last stages where the RAM blocks in the FPGA are used for the rest of the stages. Similarly, LUT-based read only memories (ROMs) are used to implement twiddle ROMs of the very last stages whereas block RAMs are used for the rest of stages[5]. The FFT is heavily pipelined to achieve as highest clock frequency as possible. Twiddle factors are generated by an external program and embedded to the VHDL code.

The implementation results after implementing in Xilinx Spartan3 FPGA (see Fig. 4) are listed in Table 1 and Table 2. Table 1 shows the implementation results whereas Table 2 shows the timing summary.

The resulting figures show that our implementation outperforms the other implementations of that kind. Its speed nearly matches that of the Xilinx core but its throughput is more than 3 times higher due to its pipeline nature.

Table 1: Implementation result

Logic utilization	Used	Available	Utilization (%)
No. of slices	3155	3584	88
No. of slice flip flops	1514	7168	21
No. of 4 input LUTs	5916	7168	82
No. of bonded IOBs	32	97	32
No. of 8×8 Multiplxers	16	16	100
No. of GCLKs	1	8	12

Table 2: Timing summary

Minimum period	10.827 (ns) (maximum frequency: 92.366 MHz)
Minimum input arrival time before clock	5.406
Maximum output required time after clock	6.216

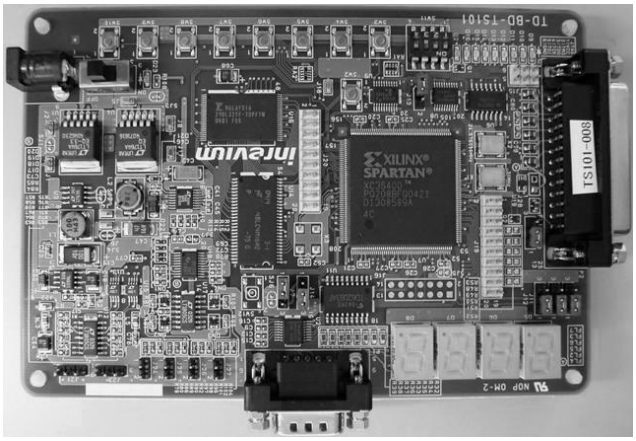


Fig. 4. Xilinx spartan 3 FPGA kit.

5. Applications of Proposed FFT in IEEE 802.16D Communication Standard

Worldwide interoperability for microwave access (WiMAX)[8],[9] is a telecommunications technology that provides wireless transmission of data using a variety of transmission modes, from point-to-multipoint links to portable and fully mobile internet access. The technology provides up to 10 Mbps broadband speed without the need for cables. The technology is based on the IEEE 802.16 standard (also called broadband wireless access). The 802.16d standard[9] uses OFDM with 256 sub-carriers.

The fundamental principle of the OFDM system is to decompose the high rate data stream (bandwidth=W) into N lower rate data streams and then to transmit them simultaneously over a large number of subcarriers[10]. The IFFT and the FFT are used for, respectively, modulating and demodulating the data constellations on the orthogonal subcarriers[11].

In an OFDM system, the transmitter and receiver blocks contain the FFT modules as shown in Fig. 5 (a) and (b). The FFT processor must finish the transform within 312.5 ns to serve the purpose in the OFDM system. Our FFT architecture effectively fits into the system since it has a minimum required time period of 10.827 ns (Table 2).

An OFDM carrier signal is the sum of a number of orthogonal sub-carriers, with baseband data on each sub-carrier being independently modulated commonly using some type of quadrature amplitude modulation (QAM) or phase-shift keying (PSK)[12]. This composite baseband signal is typically used to modulate a main RF

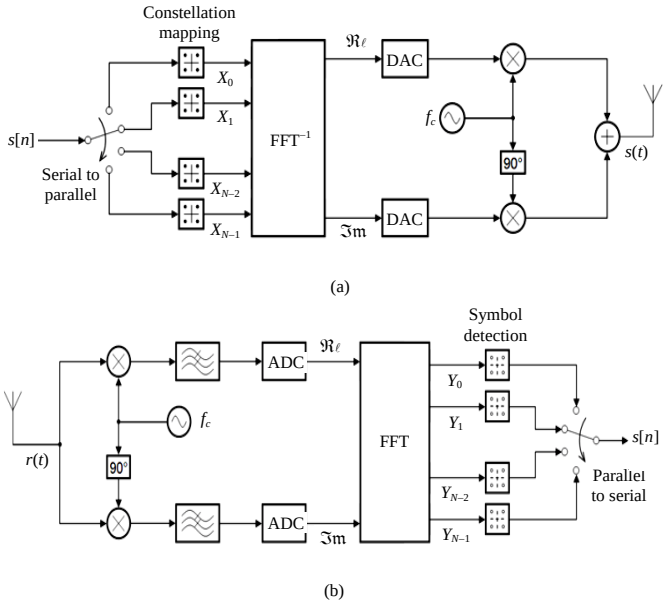


Fig. 5. OFDM module: (a) OFDM transmitter and (b) OFDM receiver.

carrier. $s[n]$ is a serial stream of binary digits. By inverse multiplexing, these are first demultiplexed into N parallel streams, and each one mapped to a (possibly complex) symbol stream using some modulation constellation (QAM, PSK, etc.). Note that the constellations may be different, so some streams may carry a higher bit-rate than others.

The receiver picks up the signal $r(t)$, which is then quadrature-mixed down to baseband using cosine and sine waves at the carrier frequency. This also creates signals centered on $2f_c$, so low-pass filters are used to reject these. The baseband signals are then sampled and digitized using analogue-to-digital converters (ADCs), and a forward FFT is used to convert back to the frequency domain[12].

6. Performance and Implementation

6.1 Hardware Requirement

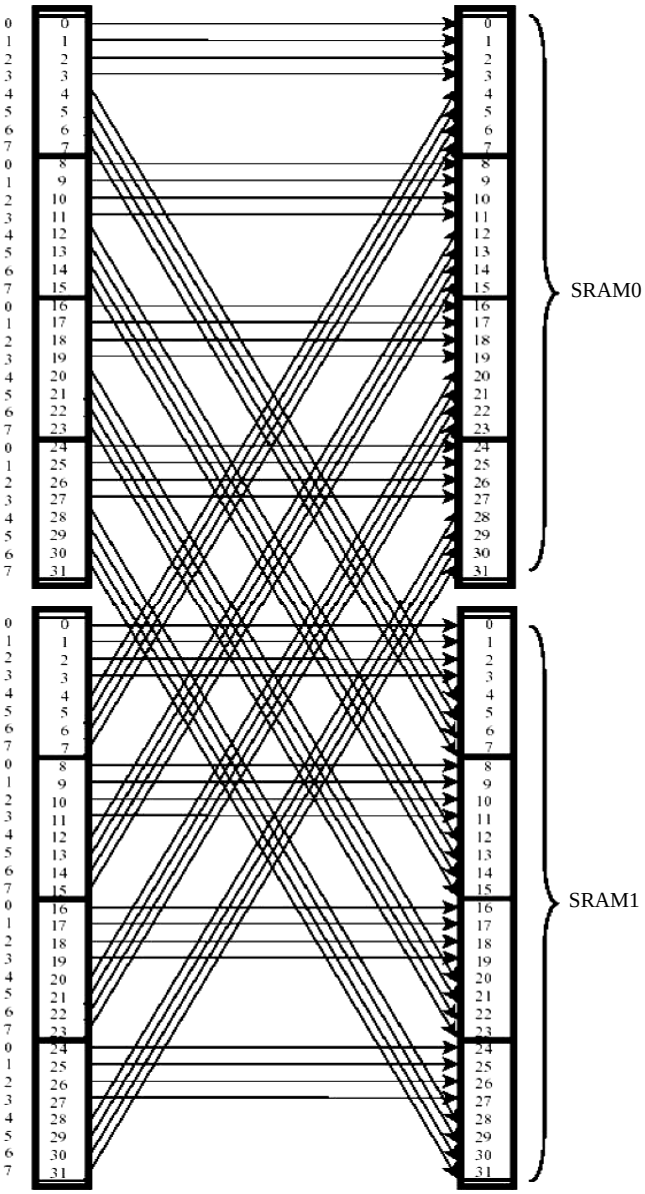


Fig. 6. Memory addressing of SRAM0 and SRAM1 for N=64.

Table 3: Hardware requirement comparison

	Multiplier #	Adder #	Memory size
R2MDC[14]	$2(\log_4 N-1)$	$4\log_4 N$	$3N/2-2$
R2SDF[15]	$2(\log_4 N-1)$	$4\log_4 N$	$N-1$
R4SDF[16]	$\log_4 N-1$	$8\log_4 N$	$N-1$
R4MDC[14]	$3(\log_4 N-1)$	$8\log_4 N$	$5N/2-4$
R4SDC[17]	$\log_4 N-1$	$3\log_4 N$	$2N-2$
Proposed	$\log_4 N-1$	$4\log_4 N$	$N-1$

The radix-4 butterfly needs 3 complex adders and 1 complex multiplier, while the proposed butterfly structure needs only 4 complex adders and 1 complex multiplier. This is because our design implements the constant multiplier by 4 reused complex adders. Fig. 6 also shows the radix-8 butterfly and radix-4 butterfly[13]. All of the above-mentioned use separated single static random access memory (SRAM) into 2 smaller SRAMs. This design can double SRAM throughput with inter-leaving access. In Table 3, the hardware requirement of the proposed design is compared with various pipelined designs.

6.2 Power Consumption

The power consumption is measured by the number of times of data transition. The data transition times are proportional to the SRAM access times. Here we assume that the adders and multipliers are active at each clock cycle because of the pipelining architecture. The more the SRAM access times are, the higher the power consumption is. Fig. 7 shows the SRAM access times versus N points FFT. The SRAM access times are linear to the number of the recursive iterations in FFT as described in (6).The SRAM is accessed twice each clock cycle, so (6) is multiplied by 2. It shows that the proposed design has less memory access than the radix-4 FFT by 20% to 40%. Therefore, the proposed architecture consumes much lower power.

$$\text{SRAM access times}=N(\text{iteration times})\times 2. \quad (6)$$

6.3 Speed

With fixed clock frequency, the processing OFDM symbol rate decreases as the FFT point N increases. A comparison with fixed clock frequency of 50 MHz is shown in Fig. 8 based on (7). It shows that the proposed architecture is better than radix-4 FFT bv 25% to 66%.

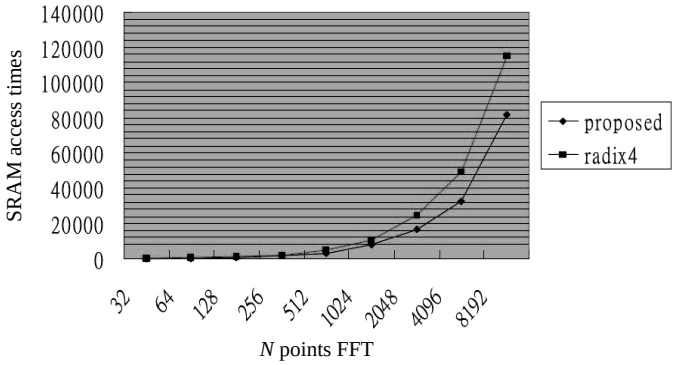


Fig. 7. SRAM access times v.s. N points FFT.

$$\text{Symbol rate}=\frac{\text{Clock frequency}\times N}{\text{Total cycles of each } N \text{ points}}. \quad (7)$$

For fixed analog-to-digital converter sampling rate, the OFDM symbol rate in receiver is fixed too. It then requires higher chip clock frequency to process higher point FFT. We make a comparison with clock frequency and *N*-points FFT as shown in (8).

$$\text{Minimum clock frequency}=\frac{\text{OFDM symbol rates}\times N}{\text{Total cycles of each } N \text{ points}}. \quad (8)$$

The proposed architecture is better than radix-4[17] FFT by 20% to 40%.

6.4 Implementation

The Fujitsu WiMAX™ SoC, MB87M3550, fully complies with the IEEE 802.16d standard using an OFDM PHY. The system-on-chip (SoC) can operate in all the available channel bandwidths from 1.75 MHz up to 20 MHz bandwidths[18]. This SoC is designed to support frequencies from 2 GHz to 11 GHz in both licensed and license-exempt bands. A programmable frequency selection generates the sample clock for any desired bandwidth. Uplink sub-channelization is supported as defined in the standard. The SoC's integrated ARM-926 RISC engine implements 802.16 upper layer MAC, scheduler, drivers, protocol stacks, and user application software. A multi-channel DMA controller handles high-speed transactions among various agents on a high performance bus.

To offload processing from the upper layer MAC and enhance performance, the Fujitsu WiMAX SoC includes a separate ARC RISC/DSP engine to execute 802.16 lower layer MAC functions. The chip's multiple hardware-based encryption/decryption engines are tightly coupled with this lower layer MAC processor and can be enabled to provide full security for the MAC privacy sub-layer.

Fig. 9 shows a simplified block diagram of the Fujitsu WiMAX SoC, MB87M3550. It shows the dual processors and main hardware blocks that implement a complete PHY-to-MAC wireless MAN solution[18]. This SoC uses OFDM PHY with 256 sub-carriers.

We have implemented the FFT, designed for efficient OFDM communication system, by 16 bits word length and synthesized in Xilinx ISE 9.1 design compiler. The maximum operation frequency is 166 MHz and it consists of 50120 gates. When the constraint of clock changes to 125 MHz, gate count reduces to 31050 gates. The gate count report does not include the memory. Our proposed design needs $N\log_4 N+9$ clock cycles to finish an *N* points FFT. The 802.16d requires 92.4 μs for 2 K points FFT and the proposed FFT work more than 89 MHz to satisfy real-time processing. For the DVB-T application, it requires 896 μs for 8 K points FFT and our design can satisfy the requirement with 46 MHz working clock.

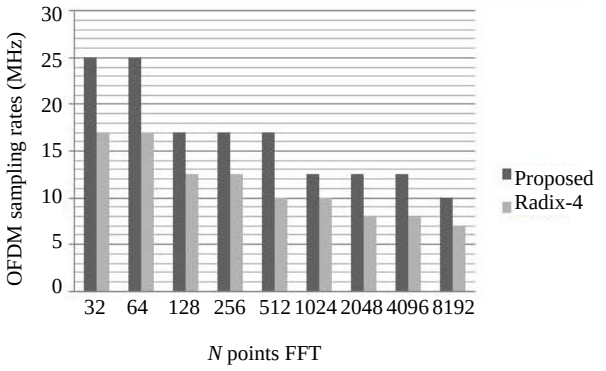


Fig. 8. OFDM symbol rates v.s. *N* points FFT.

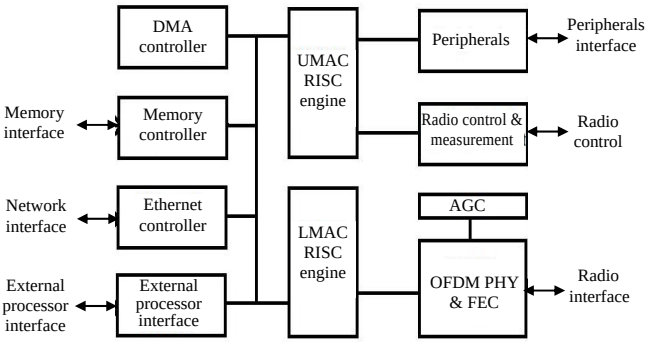


Fig. 9. Simplified block diagram of the Fujitsu WiMAX SoC.

7 Conclusions

We have proposed a memory based recursive FFT design which has much less gate counts, lower power consumption, and higher speed. The proposed architecture has three main advantages: 1) fewer butterfly iteration to reduce power consumption, 2) pipeline of radix-2^2 butterfly to speed up clock frequency, 3) even distribution of memory access to make utilization efficiency in SRAM ports.

In summary, the speed performance of our design easily satisfies most application requirements of IEEE 802.16d communication standard, which uses OFDMA modulated wireless communication system. The design uses fewer gates and hence lower cost and power consumption.

Acknowledgment

The authors express their sincere thanks for the support of the SRM University, Chennai, India, to carry out this research work. The authors are also grateful to Narayana Engineering College, Nellore, AP, India, for their constant support throughout this work.

References

[1] T. Lenart and V. Öwall, "Architectures for dynamic data scaling in 2/4/8K pipeline FFT cores," *IEEE Trans. on Very Large Scale Integration Systems*, vol. 14, no. 11, pp. 1286-1290, 2006.

[2] M. S. Minallah and G. Raja, "Real time FFT processor implementation," in *Proc. of the 2nd International Conference on Emerging Technologies*, Peshawar, Pakistan, 2006, pp. 192-195.

[3] S. Sukhsawas and K. Benkrid, "A high-level implementation of a high performance pipeline FFT on Virtex-E FPGAs," in *Proc. of the IEEE Comp. Society Annual Symp. on VLSI Emerging Trends in Systems Design*, Lafayette, LA, USA, 2004, pp. 229-232.

[4] K. Harikrishna, T. R. Rao, and V. A. Labay, "A RADIX-2^2 pipeline FFT for ultra wide band technology," in *Proc. of International Conference on Computer & Network Technology*, Chennai, India, 2009, pp. 171-175.

[5] S. He and M. Torkelson, "A new approach to pipeline FFT processor," in *Proc. of the 10th Int. Parallel Processing Symp.*, Honolulu, Hawaii, USA, 1996, pp. 766-770.

[6] J. G. Proakis and D. K. Manolakis, *Digital Signal Processing: Principles, Algorithms and Applications*, 3rd ed., Englewood Cliffs, NJ: Prentice-Hall, Inc., 1995.

[7] J. F. Wakerly, *Digital Design Principles and Practices*, 4th ed., Upper Saddle River, NJ: Pearson Education, Inc., 2006.

[8] Y. Zhang and H.-H. Chen, *Mobile WiMAX*, Boca Raton, FL: Auerbach Publications, Taylor & Francis Group, 2008.

[9] J. G. Andrews, A. Gosh, and R. Muhamed, *Fundamentals of WiMAX*, Upper Saddle River, NJ: Prentice Hall Publications, 2007.

[10] U. S. Jha and R. Prasad, *OFDM Towards Fixed and Mobile Broadband Wireless Access*, London: Artech House Inc., 2007.

[11] R. Prasad, *OFDM for Wireless Communications Systems*, London: Artech House Inc., 2004.

[12] H. Liu and G.-Q. Li, *OFDM Based Broadband Wireless Networks*, Hoboken, New Jersey: John Wiley & Sons, Inc., 2005.

[13] C.-H. Su and J.-M. Wu, "Reconfigurable FFT design for low power OFDM communication systems," in *Proc. of the 2006 IEEE Tenth International Symposium on Consumer Electronics*, St Petersburg, Russia, 2006, pp. 1-4.

[14] L. R. Rabiner and B. Gold, *Theory and Application of Digital Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, Inc., 1975.

[15] E. H. Wold and A. M. Despain, "Pipeline and parallel-pipeline FFT processors for VLSI implementation," *IEEE Trans. Comput.*, vol. 33, no. 5, pp. 414-426, 1984.

[16] A. M. Despain, "Fourier transform computer using CORDIC iterations," *IEEE Trans. Comput.*, vol. 23, no. 10, pp. 993-1001, 1974.

[17] G. Bi and E. V. Jones, "A pipelined FFT processor for word sequential data," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 12, pp. 1982-1985, 1989.

[18] U. S. Jha and R. Prasad, *OFDM Towards Fixed and Mobile Broadband Wireless Access*, London: Artech House Inc., 2007.

K. Harikrishna was born in Andhra Pradesh, India, in 1980. He is currently working as associate professor with the Department of Electronics and Computer Engineering, Narayana Engineering College, Nellore, AP, India. He received his B.Tech from Jawaharlal Nehru Technological University, Hyderabad, India in 2001, and M.S. in electrical and computer engineering from Southern Illinois University, Carbondale, IL, USA in 2004. He is a Member of International Association of Computer Science and Information Technology. He has actively attended and published various research papers in national and international conferences. He is currently pursuing Ph.D. degree with SRM University, Chennai, India.

T. Rama Rao was born in Andhra Pradesh, India, in 1972. Currently, he is working as professor and is the Head of the Telecommunications Engineering Department, Faculty of Engineering & Technology, SRM University, India. He received his Ph.D. degree in radio wave propagation studies for fixed and mobile communications over Southern India from from Sri Venkateswara University, Tirupati, India in 2000. He is a member of the IEEE, senior member of the Association of Computer, Electronics and Electrical Engineers and a member of International Association of Computer Science and Information Technology. He was the recipient of "Young Scientist" award for the XXVIth URSI (International Union of Radio Science) General Assembly, University of Toronto, Canada, held during August 1999. He has been selected as a participant to attend the "School on Data and Multimedia Communications using Terrestrial and Satellite Radio links" organized by the International Centre for Theoretical Physics), Trieste, Italy, 7-25 Feb. 2000. He worked with Aalborg University, Denmark as assistant research professor with Universidad Carlos III de Madrid, Spain and at the University of Sydney, Australia as visiting professor. Also, he worked as post-doctor research fellow with National Chio Tung Univeristy, Hsinchu, Taiwan. His research interests include radio channel measurements & modeling, broadband wireless communications and networks and wireless mesh networks. He authored papers in reputed journals and transactions and international andnational conferences.

Vladimir A. Labay was born in Winnipeg, Manitoba, Canada in 1965. He is a professor and Chair of the Department of Electrical and Computer Engineering at Gonzaga University in Spokane, Washington, USA. He earned B.Sc. and M.Sc. degrees from the University of Manitoba in 1987 and 1990, respectively. After graduating with a Ph.D. degree from the University of Victoria in 1995, he remained in Victoria, British Columbia, Canada as a lecturer and research engineer until he accepted an assistant professor position in 1999 at Eastern Washington University located in Cheney, Washington, USA. In 2007, he was a visiting professor at SRM University in Chennai, India and has previously held adjunct professorship positions at the University of Idaho, Moscow, Idaho, USA and at Washington State University, Pullman, Washington, USA. His research interests include modeling of the development of microwave/ millimeter-wave integrated circuit devices used in wireless and satellite communications.