International Journal of Electronics Communication and Computer Engineering
Volume 5, Issue 5, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Designing a 64-Point FFT/IFFT Processor for Implementation of OFDM in High Speed WLAN Applications

Rouzbeh Jahani
Department of Computer Engineering, Shahindezh Branch, Islamic Azad University, Shahindezh, Iran

Alireza Gharegozi
Department of Computer Engineering, Shahindezh Branch, Islamic Azad University, Shahindezh, Iran

Mohsen Tamaddon
Department of Computer Engineering, Shahindezh Branch, Islamic Azad University, Shahindezh, Iran

Heidar Ali Shayanfar
Department of Electrical Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran

Abstract – **In this report a methodology is presented for design of a special 32-bit 64-point processor to implement the OFDM in local wireless networks with IEEE standard 800.11a. In this FFT/IFFT, instead of direct approach, the shifter and adder is used for multiplier; thereby, it yields a major reduction in power area. In this processor a memory bank with the number of elements N=algorithm' base is considered. On this basis, the callback for digits is performed just in one stage as well as the access time to the memory is reduced.**

Keywords – **Fast Furrier Transform, Imaginary Multiply, Local Wireless Networks, Memory Bank, OFDM.**

I. Introduction

One part of sensitive communication systems is the method of data transmitting. The existing methods get a compromise between speed and the reliability. OFDM (Orthogonal Frequency Division Multiplexing) technique simultaneously brings about a very reliable and fast data transmission while the implementation is quiet difficult.

OFDM takes data transmission through some orthogonal frequencies by which method the following advantages are gained:

1. The efficient use of frequency bandwidth
2. Robustness against fading provoked by propagation along different paths
3. Relative convenient balancing because it takes place in frequency domain
4. ISI (inter symbol interference) reduction due to use of guards between samples

Figure 1 depicts the full structure of a modem for IEEE 802.11 standard. In this standard main functions and architecture of a high rate communication system are determined. Because of being a mobile system it should consume a little power. The use of multi-objective processor and various programs for each part results in more power consumption. Therefore, one can implement special hardware for major blocks of base band [1].

Most of performance in base-band transmitter/receiver systems working based on OFDM technique, refer to IFFT of transmitter and Viterbi decoder in receiver[1]. Therefore, the IFFT/FFT block is of particular significance in such systems and should be independently designed in terms of hardware. This report attends to how this block is designed in 64-point format. The FFT/IFFT processor block is widely used in high rate local wireless networks.

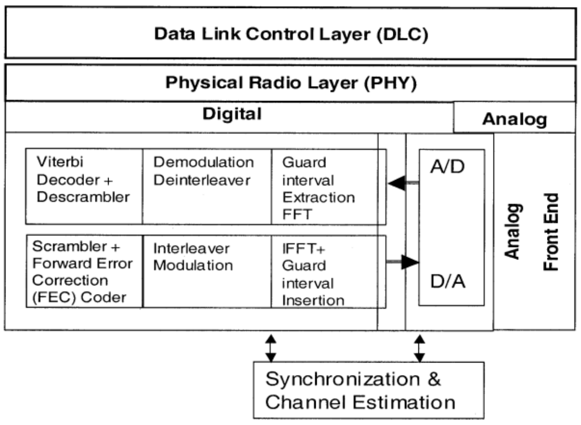
Data Link Control Layer (DLC)
Physical Radio Layer (PHY)
Digital
Analog
Viterbi Decoder + Descrambler
Demodulation Deinterleaver
Guard interval Extraction FFT
A/D
Scrambler + Forward Error Correction (FEC) Coder
Interleaver Modulation
IFFT+ Guard interval Insertion
D/A
Analog Front End
Synchronization & Channel Estimation

Fig.1. Physical layer of modem for 802.11 standard

II. FFT Algorithm

Fast furrier transform (FFT) is one of optimal algorithms for DFT calculation and in most occasions its results are the same as DFT's (with the exception of rounding error). The operation number needed for DFT is N^2 and it could be greatly reduced using FFT methods [2]. DFT is calculated by

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

Where

$$W_N=e^{-i2\pi/N} \tag{2}$$

N is a power of 2 such that $N=2^m$, and m is a natural number. This relationship can be divided to two relationships with length N/2 in a way that one of them involves odd members of x and another one utterly involves even members of x [2].

$$X(k)=\sum_{n_{even}=0}^{N-2} x(n)W_N^{nk}+\sum_{n_{odd}=0}^{N-1} x(n)W_N^{nk} \tag{3}$$

Substituting n with 2m yields [2]:

$$\begin{aligned}X(k)&=\sum_{m=0}^{N/2-2} x(2m)W_N^{2mk}+\sum_{m=0}^{N/2-1} x(2m+1)W_N^{(2m+1)k}\\&=\sum_{m=0}^{N/2-2} x(2m)\left(W_N^{2}\right)^{mk}+\sum_{m=0}^{N/2-1} x(2m+1)\left(W_N^{2}\right)^{mk}W_N^{k}\end{aligned} \tag{4}$$

W_N^2 can be simplified to

$$\begin{aligned}W_N^{2}&=\left(e^{-i2\pi/N}\right)^2\\&=e^{-i2\pi/(N/2)}=W_{N/2}\end{aligned} \tag{5}$$

So for DFT we have

$$X(k)=\sum_{m=0}^{N/2-1} x_{even}(m)W_{N/2}^{\ mk}+W_N^{\ k}\sum_{m=0}^{N/2-1} x_{odd}(m)W_{N/2}^{\ mk}\,,\quad k=0,1,\ldots,N-1 \quad (6)$$

Hence the DFT with N point changes into the DFT with N/2 [2].

$$X(k)=DFT_{N/2}\{x_{even}(m),k\}+W_N^{\ k}.DFT_{N/2}\{x_{odd}(m),k\} \quad (7)$$

Fig.2 shows how the FFT is calculated. Until now there was no simplification in operations (each element of X twice encounters N/2 operations e.g. for all X, the order o operations is N^2). The periodic form of W is of significance such that it can be shown that [2]:

$$W_N^{\ x+N/2}=W_N^{\ x}W_N^{\ N/2}$$
$$=W_N^{\ x}e^{-i\pi}=-W_N^{\ x} \quad (8)$$

In this way, one just need the half of W multiply operation. Therefore, fig. 2 will change into fig.3.

This is not the end. As mentioned, N is powered number by 2 or more so we can decompose both odd and even parts of Eq. 7 into the odd and even factors so that each part has only two members. In fig.8 this strategy is done for 8-point DFT. This decomposition is possible by $\log_2(N)$-1 times and generates $\log_2(N)$ resolution for DFT calculation. The resolution m has the number of $N/(2^{m+1}).2^m=N/2$ imaginary multiply. The final resolution becomes a 2-point DFT which can be easily calculated [2]. since each of N/2 stages of 2-point DFT involve a add and subtraction and in each resolution there is N/2 multiply function in W_N and there is $\log_2 N$ resolution, the total necessary operation number would be $N\log_2 N$ [2]. Therefore using FFT algorithm, the calculation size for high-point DFT is considerably decreased.

It can be noticed in calculation that it is possible to decrease the total constants W such that all Ws convert to the equivalent W_N. For example in fig. 3 it can be placed $W_4^0= W_2^0= W_8^0$. Fig. 5 is gained with exchanging the equivalent coefficients. Herein each resolution is decomposed to two smaller DFT; therefore, the FFT corresponds to the FFT group known as binary FFT. Also, because of time samples are recurrently divided into odd and even parts, it is known as decimation in time (DIT) [2]. If X(k) in each resolution becomes decomposed, an up duality called decimation in frequency (DIF) is gained[2].

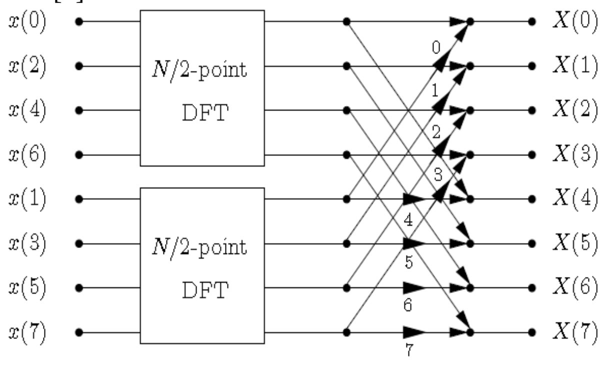

Fig.2. 8-point DFT graph for calculation of two DFT with N/2 points. Arrows indicate the multiply operation in W_8^k and numbers on the arrows denote k[2].

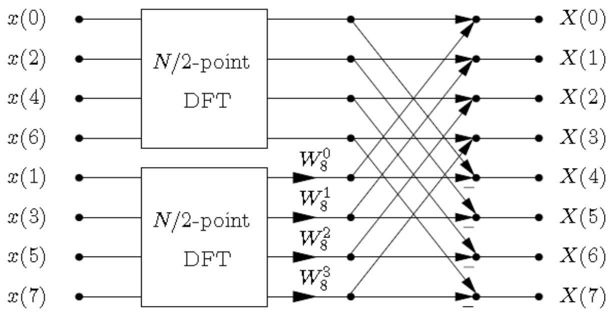

Fig.3. The modified 8-point DFT with periodic W for calculation of DFT with N/2 points. Arrows indicate multiply in W.

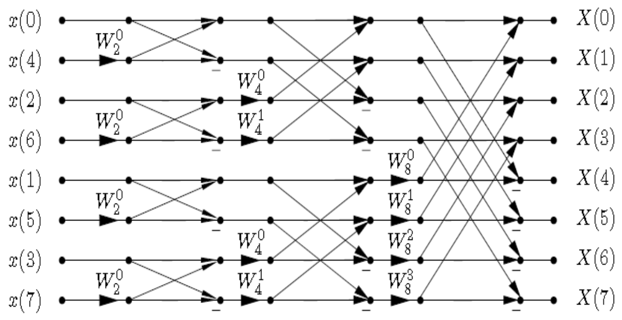

Fig.4.The completely decomposed 8-point DFT graph.

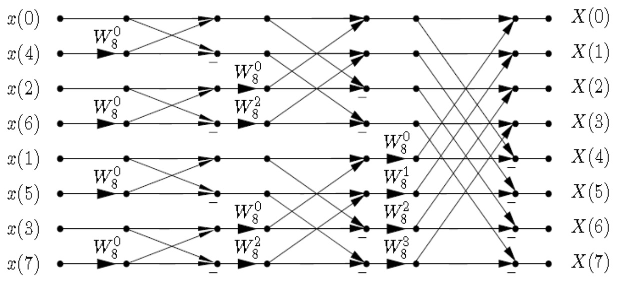

Fig.5. The completely decomposed 8-point DFT graph with equivalent W [2].

As mentioned before, our desired FFT is a 64-point one in base of 8 so considering the explanatory algorithm it can be written [1]:

$$X(s+8t)=\sum_{l=0}^{7}\left[W_{64}^{\ sl}\sum_{l=0}^{7}x(l+8m)W_8^{\ sm}\right]W_8^{\ lt} \quad (9)$$

The graph of 64-point FFT in the base of 8 is depicted in terms of DIF in fig. 6.

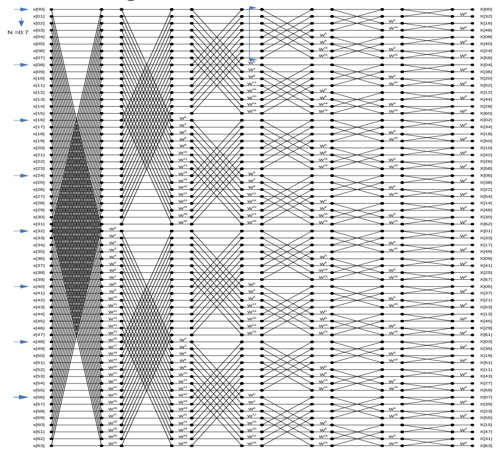
Fig.6. The graph of 64-point FFT in base 8

III. Architecture

The objective 64-point FFT/IFFT is designed considering the algorithm of base 8. The data path in FFT processor is shown in fig. 7. First, data is called from memory and finally output is written on. The bright lines indicate the control signal. The lines which are signed by D are the delayed version of original signal. Each D number denotes a unit of delay.

This FFT works in 4-level pipeline and when the calculation in 3 levels of FFT is over, the data are recorded on the memory. The data is called from and written on two distinct register banks (Bank 1 and 2). These two register banks make the implementation and calculation through the pipeline convenient [3]. The FFT/IFFT processor consists of 3 modules:

- Butterfly processor
- The address generator unit
- Control unit or MCSM (micro-coded state machine)

The butterfly processor is a conventional mathematical block for FFT calculation by which processor graphs can be easily drew and read. In fig. 8 a butterfly is shown. From fig. 8 the operations for calculation of butterfly' output is

$$A_m = A_{m-1} + B_{m-1}$$

$$B_m = (A_{m-1} - B_{m-1})W_N^{\ r} \qquad (10)$$

$$W_N^{\ r} = \exp\left(\frac{-2\pi jr}{N}\right) \qquad (11)$$

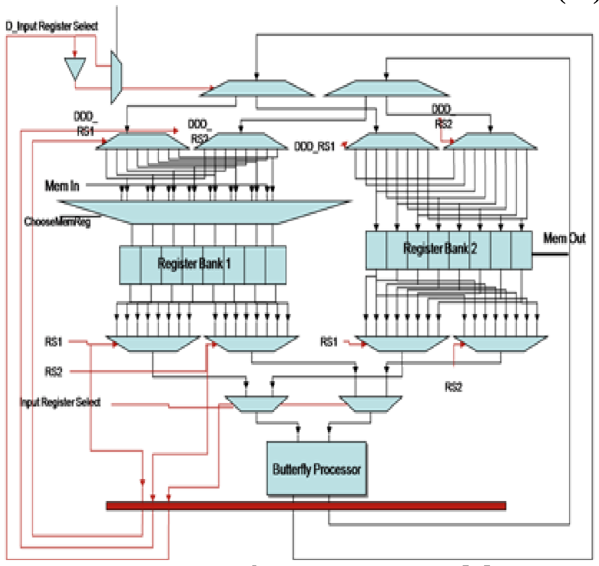

Fig.7. Data path in FFT processor [3].

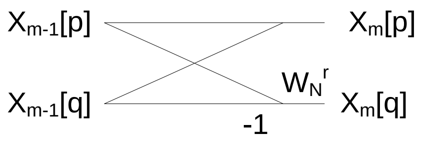

Fig.8. Butterfly cell

In fact the butterfly processor task is the calculation of imaginary operations in FFT algorithm. previous expressions suggest that this processor needs an imaginary multiplier, but the imaginary multiplier has not been implemented instead the multiply operation is done just by adders and registers. This multiply method is known as conventional signed digits (CSD) multiply [3]. However, this multiply operation ordinary for 64-point FFT with 8 base, is performed between intermediate (middle) results and the 49 intermediate (middle) non-simple constants from (Eq. 9, $W_{64}^{\ sl}, s,l \in \{1,2,...,7\}$).

The multiply operation of all intermediate constants can done with the 9 groups of constant and exchanging the real and imaginary parts as well as with selecting the appropriate sign. Since the first group of constants, e.g. (1, 0), is a simple value, just 8 groups of non-simple intermediate constants remain. Therefore, it needs to save just 8 groups (instead of 49 groups, as usual) [1].

In fig. 9 the data path in butterfly processor is presented. The multiply operation is performed by two CSD blocks which these blocks are controlled in multiplexing format. Therefore, butterfly processor can do the calculations in two levels in pipeline way.

In this way, the objective code is written in HDL language for butterfly. Since this code cloud be synthesized, using Simplicity Simplify program we could convert it to the module on gate surface (see fig.10).

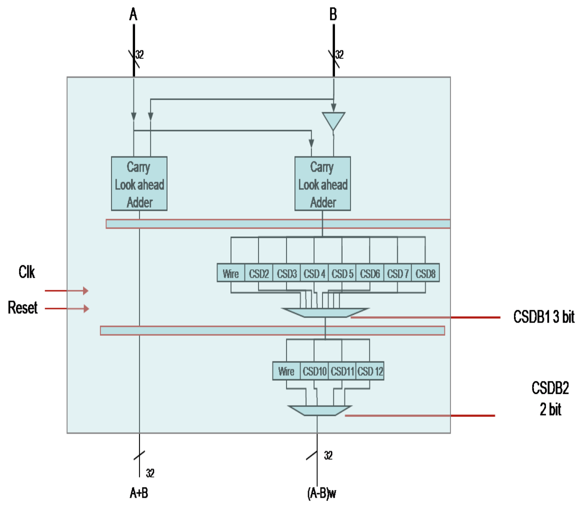

Fig.9. The data path in butterfly processor [3].

After this act, one can easily draw the circuit layout using programs such as Silicon foundry. The schematic of fig. 11 shows the gate surface of the butterfly processor. It is shown that the butterfly consists of two SCMs. The SCM is the multiplier with the 32-bit constant discriminator whose schematic is brought in fig. 11 using Simplicity program.

The address generator unit _ Using AGU, it is possible to control the address bus which goes to memory. The FFT processor is connected to 8 two-port memory banks and can simultaneously write on and written from the memory. This memory bank structure helps all information associated with the butterfly to be called back from memory [1]. If there was just on bank instead of 8 banks, the data preparation for the processor of base 8 would prolonged the more seven cycles.

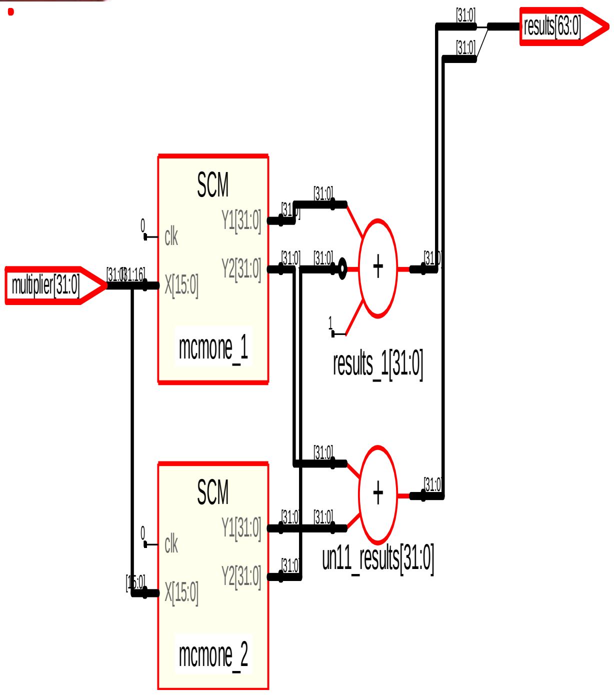

Fig.10. The gate surface circuit for butterfly designed by Synplify program

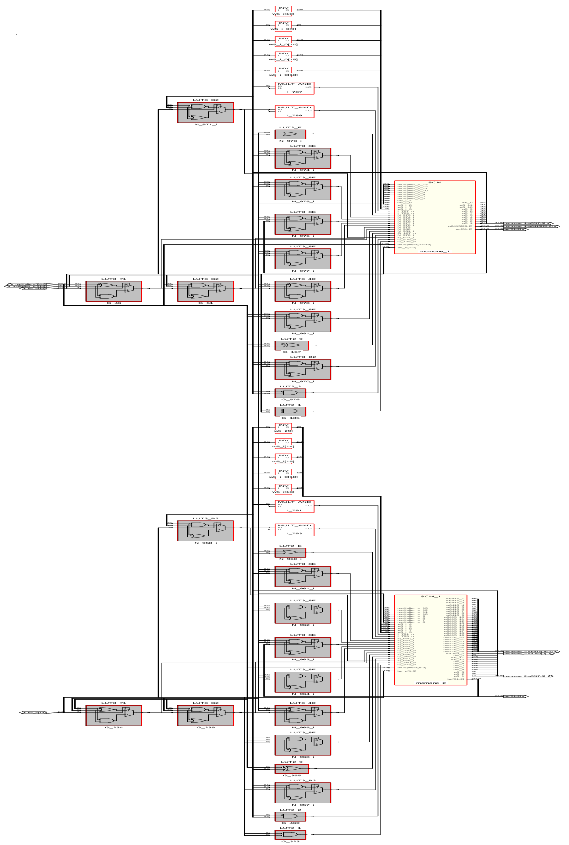

Fig.11. The gate surface circuit for multiplier with 32-bit constant discriminator designed by Simplify program

By the way, it is important to note that cycle duration for write and read is much longer than other cycles in the processor. Therefore, the presence of memory banks which are designed in FFT base, can considerable improve the performance time of the processor. Nevertheless, this strategy results in more occupation of silicon surface. The memory bank circuit on gate surface is depicted by Synplify program (fig. 12). Herein, each memory bank unit has 32-bit long. The addressing should be in an occasion in which the simultaneous write and read actions never take place. The memory has 8 address buses for read and 8 address buses for write; in other words each bank independently has address bus for write and read. In this way the address generator unit which is performed by a counter, has been made very simple and. Table 1 presents the allocation method of the memory addresses.

Control unit (micro coded state machine) _This unit saves all of the necessary control signals for the FFT processor operation. This unit uses a clock in order to generate the control state signals. The total number of generated states is 196 which can be generated via a counter [3]. Two signals of this unit make are supposed to connect with out of the FFT processor; the en_fft and done_fft. The en_ff signal clears all of the counters by which the states are generated so that the FFT calculation is restarted. The done-fft informs the other blocks that calculation is over and the output is ready. The gate surface circuit for control unit which is gained by simplifyprogram has been presented in fig. 13.

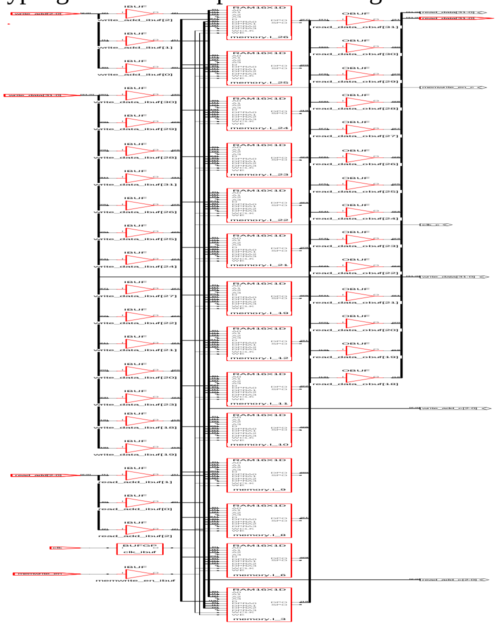

Fig.12. The memory bank in gate surface designed by Simplify program

Table 1: Memory mapping

Ban k0	Ban k1	Ban k2	Ban k3	Ban k4	Ban k5	Ban k6	Ban k7
0	1	2	3	4	5	6	7
15	8	9	10	11	12	13	14
22	23	16	17	18	19	20	21
29	30	31	24	25	26	27	28
36	37	38	39	32	33	34	35
43	44	45	46	47	40	41	42
50	51	52	53	54	55	48	49
57	58	59	60	61	62	63	56

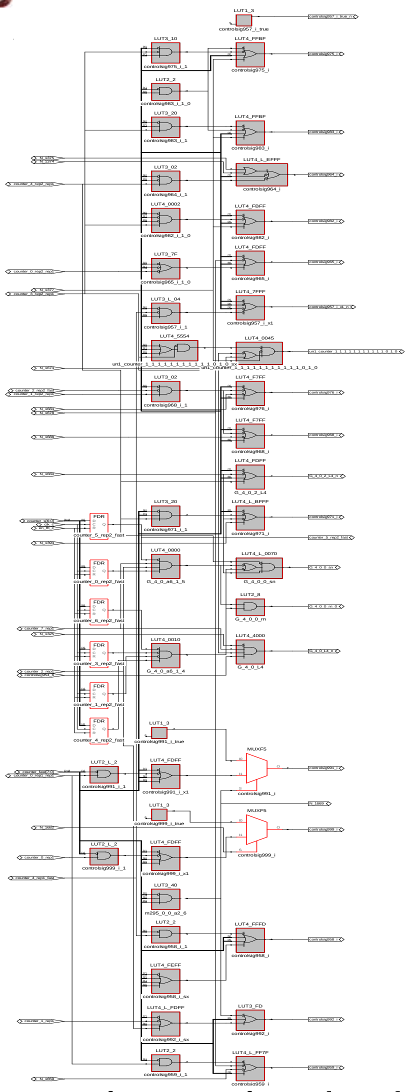
Fig.13. The gate surface circuit of control unit designed by Simplify program

IV. Measuring

The architecture was implemented by Verilog language and was simulated by Mentor Graphics' Modelsim program. The performance got approved. The Verilog -generated signals of FFT processor is presented in fig. 14.

The memory banks should by randomly initialized in order to generate such signals. The initial and final values which are registered in memory banks are brought in tables 2 and 3.

The output digits of FFT inversely appear so it should be reformed to the original state. Also, the outputs butterflies are scaled by 0.5 to prevent the calculation-originated overflow. The calculation of 64-point FFT takes 196 cycles. The processor' clock can reach to the frequency of 40 MHz which can result in the latency about. This FFT has been rewritten by core processor [3].For calculation of the IFFT in this processor, we just need to exchange the imaginary and real values of both input and outputs [1] or to make both input and output conjugated [4].

V. Conclusion

In report an OFDM-based 64-point FFT/IFFT architecture for high speed WLAN systems was explored. In order to gain a less power consumption and less silicon surface, adders and shifters were employed instead of direct imaginary multiplier. Moreover, in order to decrease the processing time and to meet the IEEE 802.11 standard, 8 memory banks were used in the FFT processor. This strategy yielded the major time reduction in access to the memory. The designed FFT processor prepares the output in 2μsec which is less than standard limit introduced in IEEE 802.11 standard [1].

Table 2: The initial values of memory banks in base 16

Bank Address	0	1	2	3	4	5	6	7
7	12153524	80010e00	80021c00	80032a00	80043800	80054600	80065400	80076200
6	c0895e81	9c598438	b8b1fc71	d50a72aa	f162e8e2	0dbb5f1b	2a13d554	466c4b8c
5	8484d609	43593986	ea58ecd4	9158a022	38585570	df5808be	8657bc0c	2d57715a
4	b1f05663	ae130c5c	18ccdf31	8386b007	ee4084dc	58fa57b1	c3b42887	2e6dfb5c
3	06b97b0d	672307ce	20330340	d942fcb2	9252f824	4b62f396	0472ef08	bd82ea7b
2	46df998d	63cc97c7	607625c0	5d1fb5ba	59c945b3	5672d3ac	531c63a6	4fc5f39f
1	b2c28465	e3132cc6	6259c1c4	e1a056c3	60e6edc1	e02d82c0	5f7419be	debaaebd
0	89375212	f9d762f3	109b9921	275fcf4e	3e24057c	54e83ba9	6bac71d7	8270a604

Table 3: Thevalues of memory bank in base 16 at the end of calculation

Bank Address	0	1	2	3	4	5	6	7
7	00a0fee9	fadafedc	045100cc	faf4035f	f7be0128	03ed03e2	02dd0a00	ee4103d1
6	fdddf7f9	ffcdf75f	fa850982	ffa8f54a	fd270bcc	f88b026f	fc9b00d9	00fbeee3
5	fb60fdbe	f697ffdf	08e4f80f	01740c13	0c9df672	050904cf	fe6dfb8f	fe5ef662
4	0c44f888	0036fbc8	fcfdfd40	f207feab	0249049f	f7d702b1	fdfb03aa	fc80fb93
3	0daefae7	0033fc55	fce50008	052cf0bb	0546fe34	0228faf4	00c30917	0a9cfe64
2	fc95f640	112f0649	fec8014f	00fe0bbc	fe58fd00	09f40336	015503ab	014c00e1
1	fca0fcb9	0445f5d2	ff7c0c7a	0264f825	06f1ffbf	eeebf967	fe6cfe1a	05bcf954
0	fe230743	0499fd51	03f40632	00f80466	f809044a	fbf9fcda	02ddfedd	052efd98

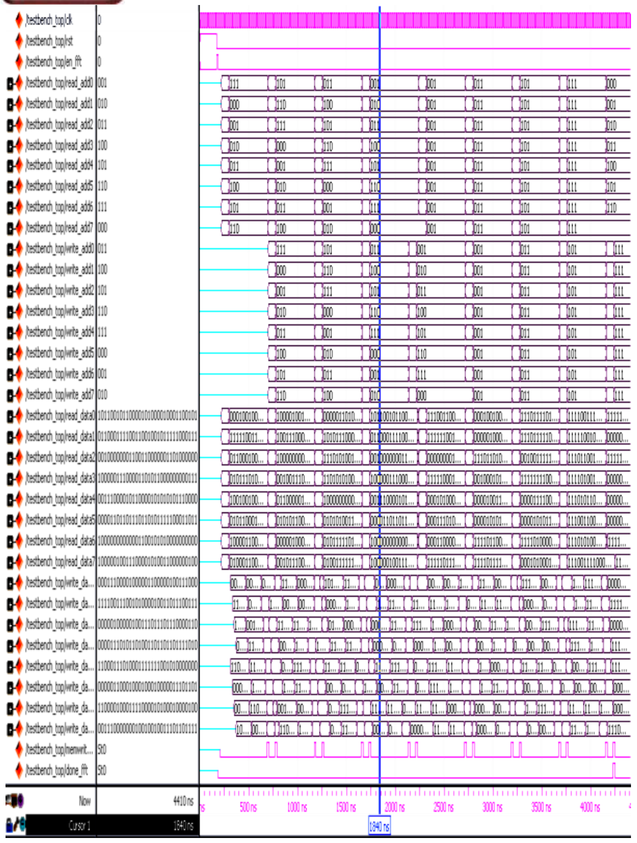
Fig.14. The generated signals simulated by Mentor Graphics' Modelsim software

REFERENCES

[1] K. Maharatna, E. Grass, U. Jagdhold, "A 64-point Fourier transform chip for high-speed wireless LAN application using OFDM", IEEE J. Solid-State Circuits, vol. 39, pp. 484-493, Mar. 2004

[2] B. M. Baas, "An approach to low-power, high-performance, fast Fourier transform processor design," Dissertation, Stanford University, Stanford, CA, 1999.

[3] Arish Alreja, "Real Time OFDM engine for High Speed Wireless Applications", GEORGIA INSTITUTE OF TECHNOLOGY, ECE 4902.

[4] Yuan Chen; Yu-Wei Lin, Chen-Yi Lee, "A Block Scaling FFT/IFFT Processor for WiMAX Applications, " Solid-State Circuits Conference, 2006. ASSCC 2006. IEEE Asian , vol., no., pp.203-206, 13-15 Nov. 2006.

[5] J. C. Candy and G. C. Temes, "*Oversampling Delta-Sigma Data Converters*," Piscataway, NJ: IEEEPress, 1992, ISBN 0-87942-285-8

[6] S. R. Northworthy, R. Schreier, and G. C. Temes, "*Delta-Sigma Data Converters*," Piscataway, NJ: IEEE Press, 1997, ISBN 0-7803-1045-4

[7] Mohammad Yavari and OmidShoaei,"*A 3.3V Second-Order Sigma-Delta Modulator for Digital Audio,"* Electrical & Computer Engineering Department, University of Tehran, Iran

[8] ShahriarRabii and Bruce A. Wooley,"*A 1.8-V Digital-Audio Sigma-Delta Modulatorin 0.8-□ m CMOS,"* Stanford University.Stanford, CA 94305

[9] SumithkumarNathany ,"*14- bit Fully differential discrete time sigma-delta modulator*,"A Thesis for the Degree of master of science Department Of Electrical Engineering KATE GLEASON College Of Engineering. Rochester Institute Of Technology .ROCHESTER, NEW YORKNOVEMBER 2006

[10] brianP.Brandt, drew E.Wingard ,Bruce A.Wooley ,"*Second Order Sigma-Delta Modulation For Digital-Audio Signal Acquisition*," IEEE JOURNAL OF SOLID STATE CIRCUITS ,26(4) ,APRIL 1991