Low-Power Memory Addressing Scheme for Fast Fourier Transform Processors

Xin Xiao, Erdal Oruklu and Jafar Saniie
Department of Electrical and Computer Engineering
Illinois Institute of Technology
Chicago, Illinois 60616
erdal@ece.iit.edu

***Abstract*— In this paper, a new memory addressing architecture is proposed for low-power radix-2 FFT implementations. Two optimization schemes are presented for dynamic power reduction. First, a multi-bank memory structure is introduced. Second, twiddle factor access times are significantly reduced with a new addressing sequence. For performance evaluation, FFT kernels with transform sizes ranging from 16 to 512 are implemented in CMOS 0.18µ technology. The synthesis results and architectural analysis indicate significant switching power reduction with no performance penalty. Power reduction factor grows with the transform size, making this architecture ideal for applications requiring long FFT operations.**

I. INTRODUCTION

Discrete Fourier transform and its realization using fast Fourier transform (FFT) exhibit a fundamental role in many recent signal processing and communications applications such as OFDM [1], spectrum analysis [2], and radar image processing [3][4]. As such, there is a clear incentive for incessant research into improving the performance of the FFT processors in portable, compact devices in order to facilitate higher throughput and lower power consumption.

A typical FFT processor is composed of butterfly calculation units, an address generator, and memories for computational data and the twiddle factors. Often FFT processors use only one butterfly unit and the "in-place" strategy, which require the least amount of hardware resources. We only consider the one butterfly structure and "in-place" addressing scheme in this paper.

In recent years, several new approaches and algorithms have been proposed to improve the performance of a FFT processor. Cohen [5] proposed a simplified control logic for radix-2 FFT processing. Based on Cohen's method, Ma's [6] algorithm can speedup the radix-2 FFT operation by reducing the address generation delay. Xiao et al. [7] developed a new butterfly structure which leads to an even faster and simpler control logic for radix-2 FFT.

In addition, researchers focused on the low power design of FFT processors: Ma et al. [8,9] proposed methods to improve the memory accessing speed and reduce the power consumption. Hasan et al. [10] proposed to reduce the power consumption of short-length radix-4 FFTs. Gate-level algorithms [11,12] have been also proposed to reduce the FFT processor's power consumption by lower supply voltage techniques and/or voltage scaling.

This paper presents a new FFT data and coefficient addressing method based on algorithm given in [7] in order to reduce switching activity and dynamic power consumption. In general, dynamic power consumption in CMOS circuits can be characterized by the following equation:

$$P_{dynamic} = \alpha \cdot C_{total} \cdot V_{DD}^2 \cdot f \qquad (1)$$

where α is the switching activity, V_{DD} is the supply voltage, f is the frequency and C_{total} is the total switching capacitance charging and discharging in the circuit. In this work, we focus on architectural techniques to reduce two parameters in (1), α and C_{total}. With the proposed scheme, a multi-bank memory structure is used for data memory accesses, resulting in reduced overall capacitance load on the SRAM bit-lines. Furthermore, a new butterfly calculation order reduces the access times for twiddle factors and minimizes the switching activity. Any other gate-level techniques such as such as integrating sleep transistors, dual-V_{th} transistors or multiple supply voltages can be also implemented for further power optimization, in particular for leakage current reduction.

In the following sections, first, radix-2 FFT operations and the signal flow graph are discussed. Then, the new FFT addressing scheme is described and compared to the existing methods. Finally, synthesis results are presented and analyzed.

II. RADIX-2 FFT AND ADDRESS GENERATION LOGIC

The N-point discrete Fourier transform is defined by

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \qquad k = 0,1,\ldots, N-1, \quad W_N^{nk} = e^{-j\frac{2\pi}{N}nk} \qquad (2)$$

Fig. 1 shows the signal flow graph of a 16-point decimation-in-frequency (DIF) radix-2 FFT. For N-point FFT, there are log_2N passes and each pass contain $N/2$ butterfly operations. Each crossed line in Fig.1 represents a butterfly operation. Fig. 2 shows a radix-2 FFT butterfly structure at pass m. Equations (3) and (4) describe the butterfly calculations shown in Fig. 2.

$$x_{m+1}(p) = x_m(p) + x_m(q) \qquad (3)$$

$$x_{m+1}(q) = [x_m(p) - x_m(q)]W_N^r \qquad (4)$$

Parallel and "in-place" butterfly operations require reading two inputs and writing two outputs in each clock cycle. Two memory banks are utilized to realize this property. In this case,

978-1-4244-4480-9/09/$25.00 ©2009 IEEE

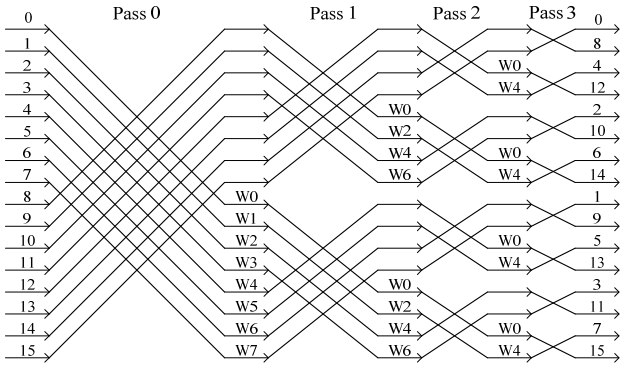

Fig. 1. Signal flow graph of 16-point FFT

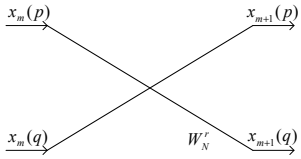

Fig. 2. Butterfly unit at pass m

a special addressing scheme is needed to prevent the conflicting addresses. Several addressing algorithms [5-7,13] have been proposed to realize the conflict free data accessing. The contribution of this paper is to design a new FFT processor which reduces the power consumption by using multi-bank memories and reducing twiddle factor accessing times.

In most FFT processors, the twiddle factors are stored in a ROM. Normally, the twiddle factor ROM will be accessed once at each clock cycle. There are *N/2* twiddle factors in an *N*-point FFT processing, but an *N/4*-point twiddle factor pattern is widely used in FFT processors. It is based on the symmetrical property of the twiddle factors. *N/2*-point twiddle factors are W_0 to $W_{\frac{N}{2}-1}$. Assume that two twiddle factors are $W_x = R_x + jI_x$ and $W_y = R_y + jI_y$. When $y = x + N/4$, we have $R_y = -I_x$ and $I_y = R_x$ due to trigonometric properties. Hence, $W_y = -I_x + jR_x$, so the twiddle factor ROM needs to store only W_0 to $W_{\frac{N}{4}-1}$. The remaining $W_{\frac{N}{4}}$ to $W_{\frac{N}{2}-1}$ can be obtained by simple derivations.

III. Proposed New Addressing Scheme

In [7], a hardware-efficient FFT engine with reduced addressing logic was introduced by using a butterfly structure that modifies the conventional one by adding exchange control circuits at the input and output of the butterfly. With this architecture, the two inputs and two outputs of any butterfly can be exchanged; hence, all data and addresses in FFT processing can be reordered. Using this flexible input and output ordering, a reduced addressing logic is designed that does not need parity check operations and it is "in-place". Fig. 3 shows the signal flow graph of a 16-point FFT using the algorithm given in [7]. In Fig. 3, the butterfly inputs or outputs indicated by dashed lines denote that the data have been exchanged. In this study, we use the same butterfly

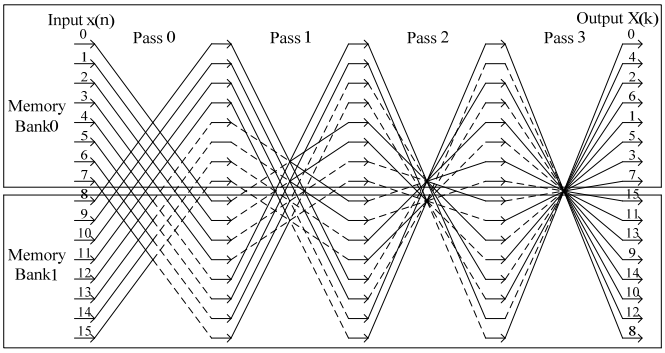

Fig. 3. Signal flow graph of 16-point FFT based on [7].

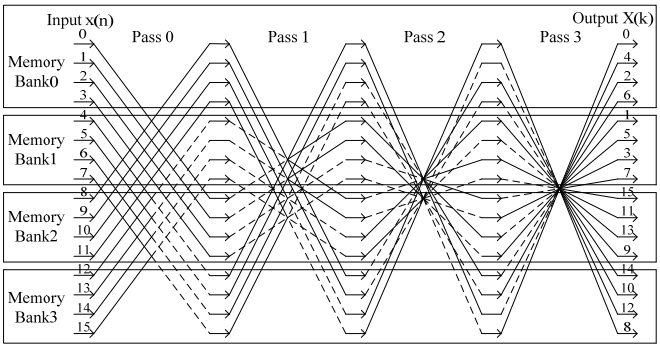

Fig. 4. Signal flow graph of 16-point FFT using proposed method

input/output exchange logic and propose a new addressing scheme for reduced power dissipation. Power reduction is achieved due to two methods introduced to the FFT memory addressing scheme. First method is memory bank partitioning and second method is reordering of the coefficient access sequence. The signal flow graph and the address generation table of the proposed method are shown in Fig. 4 and Table I, respectively.

A. Memory Bank Partitioning

Since FFT operation largely consists of data and coefficient memory accesses, it is desirable to reduce the power dissipation caused by memory accesses. Memory bank partitioning and bitline segmentation is an important technique to reduce the power dissipation in RAMs. The bitlines (each read and write port is associated with one bitline) in the SRAM logic are a significant source of energy dissipation due to the large capacitive load. This capacitance has two components, wire capacitance of the bitlines and the diffusion capacitance of each pass transistor connecting bitline to bitcells. Hence, the capacitive load increases linearly with the components attached to the bitline i.e., the number of words or size of the memory.

In the proposed scheme, the data memory is partitioned into four memory banks instead of two. At any instant, only half of the memory banks are active As a result, the capacitive loading in each memory bank is lowered since the bitline wire length and the number of pass transistors connected to the bitline are one fourth of the original bitline. The first two memory banks, *bank0* and *bank1* are accessed by the upper leg of the butterfly structure, and *bank2* and *bank3* are accessed by the lower leg of the butterfly (see Fig.

TABLE I. ADDRESS GENERATION TABLE OF THE PROPOSED METHOD FOR 32-POINT FFT

Counter $B(b_3b_2b_1b_0)$	Pass 0			Pass 1		
	Bank0,1address $b_3b_2b_1b_0$	Twiddle factor address $b_2b_1b_0$	Bank2,3 address $b_3b_2b_1b_0$	Bank0,1 address $b_3b_0b_2b_1$	Twiddle factor address $b_2b_1 0$	Bank2,3 address $\bar{b}_3b_0b_2b_1$
0000	0000	000	0000	0000	000	1000
0001	0001	001	0001	0100	000	1100
0010	0010	010	0010	0001	010	1001
0011	0011	011	0011	0101	010	1101
0100	0100	100	0100	0010	100	1010
0101	0101	101	0101	0110	100	1110
0110	0110	110	0110	0011	110	1011
0111	0111	111	0111	0111	110	1111
1000	1000	000	1000	1000	000	0000
1001	1001	001	1001	1100	000	0100
1010	1010	010	1010	1001	010	0001
1011	1011	011	1011	1101	010	0101
1100	1100	100	1100	1010	100	0010
1101	1101	101	1101	1110	100	0110
1110	1110	110	1110	1011	110	0011
1111	1111	111	1111	1111	110	0111

Pass 2			Pass 3			Pass 4		
Bank0,1 address $b_3b_1b_0b_2$	Twiddle factor address $b_2 00$	Bank2,3address $\bar{b}_3\bar{b}_1b_0b_2$	Bank0,1address $b_3b_2b_1b_0$	Twiddle factor address 000	Bank2,3 address $\bar{b}_3\bar{b}_2\bar{b}_1b_0$	Bank0,1address $b_3b_0b_2b_1$	Twiddle factor Address 000	Bank2,3address $\bar{b}_3b_0\bar{b}_2b_1$
0000	000	1100	0000	000	1110	0000	000	1111
0010	000	1110	0001	000	1111	0100	000	1011
0100	000	1000	0010	000	1100	0001	000	1110
0110	000	1010	0011	000	1101	0101	000	1010
0001	100	1101	0100	000	1010	0010	000	1101
0011	100	1111	0101	000	1011	0110	000	1001
0101	100	1001	0110	000	1000	0011	000	1100
0111	100	1011	0111	000	1001	0111	000	1000
1000	000	0100	1000	000	0110	1000	000	0111
1010	000	0110	1001	000	0111	1100	000	0011
1100	000	0000	1010	000	0100	1001	000	0110
1110	000	0010	1011	000	0101	1101	000	0010
1001	100	0101	1100	000	0010	1010	000	0101
1011	100	0111	1101	000	0011	1110	000	0001
1101	100	0001	1110	000	0000	1011	000	0100
1111	100	0011	1111	000	0001	1111	000	0000

4). The most significant bit (MSB) of the addresses (counter) (b_3 in Table I) determine which two memory banks will be accessed; the remaining two memory banks will be inactive. A major advantage of this scheme is that the memory bank switching occurs only *once* in the middle of a pass. In the first half of the pass, same two memory banks are used and in the second half of the pass, the other two memory banks are accessed. There is no precharging and discharging of bitlines in the inactive memory banks for half the cycle of a pass.

B. Reordering Coefficient Access Sequence

A new coefficient (twiddle factor) access sequence is introduced in this study. For all butterfly passes (except *pass 0)*, the twiddle factor addresses are ordered in such a way that the coefficients using the same address are grouped together and accessed sequentially. Therefore, if a register is used to store the twiddle factor temporarily, we don't have to access the twiddle factor ROM every clock cycle. For example, in *pass 1* in Table I, only 8 accesses are needed instead of 16, and in *pass 2,* only 4 accesses instead of 8 and so on.

Equations (5) and (6) show the twiddle factor access times of the existing method given in [9] and the proposed method for $N = 2^r$ point FFT.

$$\text{Existing method: } \frac{N}{2}\times(r-2)+2=\frac{N}{2}\left((\log_2 N)-2\right)+2 \quad (5)$$

$$\text{Proposed method: } \sum_{i=2}^{r-1}2^i+2=2^r-2=N-2 \quad (6)$$

TABLE II. REDUCTION IN TWIDDLE FACTOR ACCESSES

	16-point FFT	32-point FFT	64-point FFT	128-point FFT	256-point FFT	512-point FFT
FFT scheme given in [9]	18	50	130	322	770	1794
Proposed FFT scheme	14	30	62	126	254	510
Reduction	22%	40%	53%	61%	66%	72%

Table II shows the twiddle factor access times of several different length FFTs.

C. Implementation

To implement an $N = 2^r$ point FFT by the proposed design, first, we need a *(r-1)*-bit *Butterfly Counter* $B = b_{r-2}b_{r-3}...b_1b_0$, and a $\lceil \log_2 r \rceil$- bit *Pass Counter* $P = (r-1), ... ,2,1,0$. In practice, *Pass Counter P* and *Butterfly Counter B* can be combined to a single *counter D*, where B is the least significant *(r-1)* bits of *counter D,* and P is the most significant $\lceil \log_2 r \rceil$ bits of *counter D*. In addition, we need one *(r-2)*-bit barrel shifter: assume $RR(x_u x_{u-1} x_{u-2} ... x_1 x_0, v)$ indicates rotate-right counter $x_u x_{u-1} x_{u-2} ... x_1 x_0$ by v bit. At *pass s,* the read and write addresses of the upper legs of the butterfly is $A_u = RR(b_{r-3} ... b_1 b_0, s) = a_{r-3}a_{r-4} ... a_1 a_0$, and b_{r-2} decides if *bank0* or *bank1* will be accessed.

For example, for the 32-point FFT shown in Table I, at *pass 2*, the address of the upper legs of the butterfly is $RR(b_2 b_1 b_0, 2) = b_1 b_0 b_2$, and when b_3=0, memory *bank0* will be accessed, when b_3=*1*, memory *bank1* will be accessed. For the read and write addresses of the lower legs of the butterfly, *(r-2)* inverters are needed, the address is given by $\bar{a}_{r-3}\bar{a}_{r-4} ... \bar{a}_{r-s-1}a_{r-s-2} ... a_1 a_0$, and b_{r-2} decides if *bank2* or *bank3* will be accessed at *pass 0*. When $b_{r-2} = 0$, *bank2* will be accessed, when $b_{r-2} = 1$, *bank3* will be accessed, for other passes $b_{r-2} = 0$ means *bank3* will be accessed, $b_{r-2} = 1$ means *bank2* will be accessed. The address of twiddle factors is given by $a_{r-s-3} ... a_0 0...0$ (s '0's).

IV. RESULTS

The proposed FFT algorithm is synthesized using TSMC CMOS 0.18µm technology. Synthesis is performed with Synopsys *Design Compiler* tool. The target clock frequency was 200MHz for all designs. Table III shows the dynamic switching power results obtained by Synopsys *Power Compiler* for FFT transform sizes ranging from 16 to 512. The proposed FFT architecture is compared to the existing FFT architecture given in [7]. It can be seen that, switching activity reduces significantly (more than 70% for >512-point FFT operations). Considering also the contribution of bitline segmentation, the results closely match with the analytical study given in Table II. The area and speed results are identical to the previous implementation given in [7].

TABLE III. SYNTHESIS RESULTS - REDUCTION IN SWITCHING POWER (in mW)

	16-point FFT	32-point FFT	64-point FFT	128-point FFT	256-point FFT	512-point FFT
FFT scheme given in [9]	2.91	3.96	6.27	10.77	20.11	38.19
Proposed FFT scheme	1.94	2.12	2.7	3.89	6.90	11.38
Reduction	33%	47%	57%	64%	66%	71%

V. CONCLUSION

This paper proposed a new addressing scheme for FFT processors using memory bank partitioning and reducing the access times of the twiddle factors. The synthesis results demonstrate that switching power reduction grows with the transform size, making this architecture ideal for applications requiring long FFT operations.

REFERENCES

[1] R. M. Jiang, "An area-efficient FFT architecture for OFDM digital video broadcasting," *IEEE Trans. Consum. Electron*., vol. 53, no.4, pp. 1322-1326, Nov. 2007.

[2] H. Xiaohong, W. Zhaohua, and H. Guoqiang, "New method of estimation of phase, amplitude, and frequency based on all phase FFT spectrum analysis", *Intelligent Signal Processing and Communication Systems, ISPACS 2007*, pp. 284-287, Nov.2007.

[3] K. Natroshvili, O. Loffeld, H. Nies, A.M. Ortiz, and S. Knedlik, "Focusing of general bistatic SAR configuration data with 2-D inverse scaled FFT", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 10, pp. 2718-2727, Oct. 2006.

[4] E. Cetin, R.C.S. Morling, and I. Kale, "An integrated 256-point complex FFT processor for real-time spectrum analysis and measurement", *IEEE Instrumentation and Measurement Technology Conference, IMTC/97*, vol. 1, pp. 96-101, May 1997.

[5] D. Cohen. "Simplified control of FFT hardware", *IEEE Trans. Acoust., Speech, Signal Process*, vol. 24, pp. 577–579, Dec. 1976.

[6] Y.Ma, "An effective memory addressing scheme for FFT processors", *IEEE Trans. on Signal Process*., vol. 47, no. 3, pp. 907–911, March 1999.

[7] Xin Xiao, E. Oruklu, and Jafar Saniie; "An efficient FFT engine with reduced addressing logic"; *IEEE Transactions on Circuits and Systems II: Express Briefs,* vol. 55, no. 11, pp. 1149-1153, Nov. 2008.

[8] Y. Ma, and L. Wanhammar,"A hardware efficient control of memory addressing for high-performance FFT processors", *IEEE Transactions on Signal Processing*, vol. 48, no. 3, pp. 917-921, March 2000.

[9] Y. Ma, and L. Wanhammar,"A coefficient access control for low power FFT processors", *IEEE 42nd Midwest Symposium on Circuits and Systems*, vol.1, pp. 512-514, Aug. 1999.

[10] M. Hasan, T. Arslan, and J.S. Thompson, "A novel coefficient ordering based low power pipelined radix-4 FFT processor for wireless LAN applications", *IEEE Transactions on Consumer Electronics*, vol. 49, no.1, pp. 128-134, Feb. 2003.

[11] M.S. Zainal, S. Yoshizawa, and Y. Miyanaga, "Low power FFT design for wireless communication systems", *International Symposium on Intelligent Signal Processing and Communications Systems ISPACS 2008*, pp. 1-4, Feb. 2009.

[12] S. Saponara, L. Serafini, and L. Fanucci, "Low-power FFT/IFFT VLSI macro cell for scalable broadband VDSL modem", *The 3rd IEEE International Workshop on System-on-Chip for Real-Time Applications, pp.*161-166, June 2003.

[13] L.G. Johnson, "Conflict free memory addressing for dedicated FFT hardware," *IEEE Trans. Circuits Syst. II*, vol. 39, pp. 312-316, May 1992.