第18卷 第1期
2008年1月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol. 18 No. 1
Jan. 2008

应用于时频分析的低功耗短时傅里叶变换处理器

张师群，于敦山，盛世敏
（北京大学 微电子系，北京 100871）

摘 要：提出了一个应用于时频分析的短时傅里叶变换处理器。为了克服已有的离散短时傅里叶变换算法和结构的缺点，给出了一种基于快速傅里叶变换阵列的新结构。根据实际需要提出了一种新的高频域分辨率的SDF(Single－path Delay Feedback)结构FFT单元，和传统的SDF结构FFT单元相比，反馈FIFO的深度和蝶形单元的数量都有所降低。再加上开发窗函数的对称性和适当合并硬件资源，与原始设计相比处理器的功耗降低了20%。使用中芯国际0.18微米工艺实现之后，系统工作时钟可以达到200MHz，即该处理器可以满足同样频率的采样信号的实时时频分析需求。

关键词：短时傅里叶变换；时频分析；低功耗
中图分类号：TN4　　**文献标识码**：A　　**文章编号**：1673－629X(2008)01－0001－06

A Low Power Short Time Fourier Transform Processor for Time－Frequency Analysis

ZHANG Shi-qun, YU Dun-shan, SHENG Shi-min
(Institute of Microelectronics, Peking University, Beijing 100871, China)

Abstract: A discrete short time Fourier transform (STFT) processor for time－frequency analysis is proposed in this paper. To overcome the drawbacks of previous discrete STFT algorithms and architectures, a new architecture based on fast Fourier transform (FFT) array is presented. An optimized single－path delay feedback (SDF) FFT architecture is proposed for high frequency resolution, which reduces feedback FIFO depth and butterfly units' number in comparison with the traditional one. Along with exploiting the symmetry of window function and performing hardware combination, power consumption decreased by 20%. The processor has been implemented with SMIC 0.18μm standard cell library, static timing analysis and dynamic simulation with timing back－annotated show that under worst－case conditions, 200MHz clock rate can be achieved, which means that the real－time time－frequency analysis demand can be fulfilled under the same sample frequency.

Key words: short time Fourier transform; time－frequency analysis; low power
CLC number: TN4　　**Document code**: A　　**Article ID**: 1673－629X(2008)01－0001－06

0 Introduction

Discrete Fourier Transform (DFT) is the standard method for spectrum analysis in digital signal processing field, usually implemented with Fast Fourier Transform (FFT)[1]. It projects signals in the whole time domain on e^{jwn}, so DFT works smoothly when employed for finite－length signal composed of sinusoidal components as long as each of the sinusoidal components is stationary. However, there are many practical situations where the signal to be analyzed is non－stationary, such as speech, radar, sonar signals. Applying simple DFT on the complete signal will provide misleading results[2].

To get around the time dependent character of the signal, time－frequency analysis was brought up in 1940s, and developed tremendously since 1980s. Many time－frequency analysis methods are proposed, including Short－Time－Fourier－Transform (STFT), Wavelet Transform(WT), Wigner－Ville Distribution (WVD) and Cohen's Class[3]. STFT is the most direct way to adapt DFT to non－stationary signals. It segments the sequence into a set of subsequences of short length, with each subsequence centered at uniform intervals of time which approximates stationary signal, so DFT can be applied separately on these subsequences[4]. STFT, also

收稿日期：2007－05－22
基金项目：国家863计划项目(2003AA1Z)
作者简介：张师群(1980－)，男，河北人，博士研究生，研究方向为SoC设计与设计方法学；于敦山，教授，研究方向为SoC设计与设计方法学；盛世敏，研究员，研究方向为SoC设计与设计方法学。

known as the Time – dependent Fourier Transform, is defined as

$$X_{STFT}(e^{jw}, n) = \sum_{-\infty}^{\infty} x[n+m]w[m]e^{-jwm} \quad (1)$$

where $w[m]$ is the properly chosen window factor. X_{STFT} is a function of two separate variables, and the display of the magnitude of the STFT is usually referred as the spectrogram, in which time dependent spectrum of $x[n]$ can be easily analyzed. Usually, for digital processing demand, the spectrum is usually calculated discretely after sampling X_{STFT} in frequency domain, and each time – dependent unit can be calculated as a separate DFT as follows:

$$X_{STFT}[k,n] = X_{STFT}(e^{j2\pi k/N}, n) =$$
$$\sum_{m=0}^{R-1} x[n+m]w[m]e^{-j2\pi km/N} =$$
$$\sum_{m=0}^{R-1} x[n+m]w[m]W_N^m, 0 \leqslant k \leqslant N-1 \quad (2)$$

X_{STFT} can also be sampled in time domain for $n = lL$, in which $0 \leqslant L \leqslant R$, $-\infty \leqslant l \leqslant +\infty$. Original signal can be reconstructed uniquely provided $L \leqslant R \leqslant N$.

STFT is the simplest time – frequency analysis method, and has the same frequency resolution in high frequency region as in low region which is important to distinguish signals with constant frequency difference. Moreover, equation of STFT is regular enough for hardware implementation.

A low power STFT processor based on optimized Radix – 2^x SDF FFT processor array for high frequency domain resolution is proposed in this paper, which can fulfill the real – time processing demand for complex signals with sample frequency the same as the system clock. To simplify the question and not lose universality, in this paper, these values are chosen so that $N = 2, R = 32, L = 3$ for algorithm and architecture analysis. In the following sections, previous works are briefly reviewed and the main drawback of them is addressed.

Then a new architecture based on FFT arrays for discrete STFT processing is proposed. Optimizations for hardware efficiency and power consumption are also presented in this paper.

1 Previous Work on STFT

Before Cooley – Tukey algorithm was invented in 1965, DFT was calculated directly. The transform function of direct calculation and Goertzel algorithm for calculating the kth bin of DFT is shown as follows[5]:

$$H_k(z) = \frac{1}{1 - W^{-k}z^{-1}} = \frac{(1 - W^k z^{-1})}{(1 - W^k z^{-1})(1 - W^{-k}z^{-1})}$$
$$= \frac{(1 - W^k z^{-1})}{1 - 2\cos\left(\frac{2\pi}{N}k\right)z^{-1} + z^{-2}} \quad (3)$$

Block diagram for calculating one DFT bin using direct DFT and Goertzel algorithm are shown in figure 1 and figure 2 respectivily. Because the multiplication on feed forward path should be performed only once per N cycles, the computational complexity of Goertzel algorithm reduced from $3N$ multiplications and $5N$ additions to $2N + 3$ multiplications and $4N + 2$ additions for complex input, in comparison with direct DFT calculation.

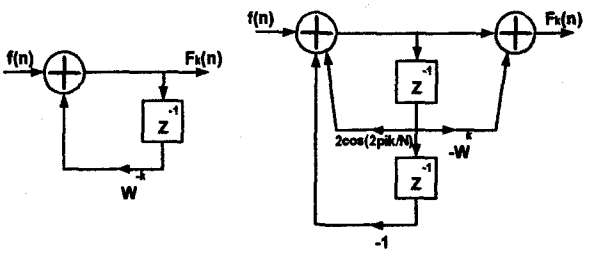

Figure1 direct DFT Figure2 Goertzel algorithm

These IIR filter implementations for DFT are hardware consumptive and replaced in most applications after Cooley – Tukey algorithm invented in 1965, which is still the dominated one for DFT implementation now[1]. However, when the concerned frequency band is narrow enough, the Goertzel algorithm might have computational advantage than Cooley – Tukey algorithm. On the other hand, when calculating discrete STFT sample by sample with rectangular window, Goertzel algorithm shows its superiority. The transform function of kth bin of discrete STFT is[6,7]

$$H_k(z) = \frac{1 - z^{-N}}{1 - W^{-k}z^{-1}} = \frac{(1 - z^{-N})(1 - W^k z^{-1})}{(1 - W^k z^{-1})(1 - W^{-k}z^{-1})}$$
$$= \frac{(1 - z^{-N})(1 - W^k z^{-1})}{1 - 2\cos\left(\frac{2\pi}{N}k\right)z^{-1} + z^{-2}} \quad (4)$$

Therefore, filter bank constructed with units shown in figure 1 and figure 2 was proposed for discrete STFT computation, and they are named Sliding DFT and Sliding Goertzel DFT, as shown in figure 3 and figure 4. Note that data from the same comb filter can be broadcast to all N IIR filters, which means only one comb filter is required.

Only rectangular and Hanning window can be applied

efficiently under these circumstances, which is the main drawback of these methods, or the simplicity of these structure will be compromised by time domain multiplications[7]. However, in most cases, window functions play critical roles to alleviate the Gibbs phenomenon, STFT processors should work smoothly for arbitrary window function according to the property of signals for time – frequency analysis.

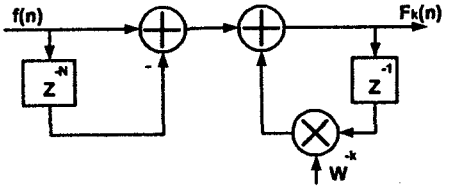

Figure3　Sliding DFT

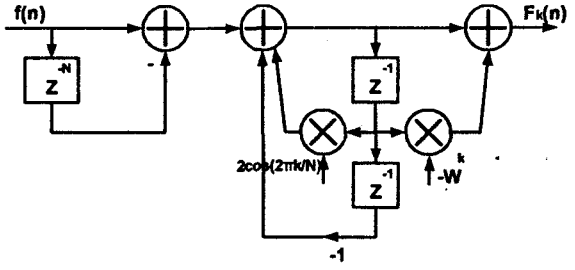

Figure4　Sliding Goertzel

2　FFT Processor with High Frequency Resolution

Cooley – Tukey algorithm uses divide – and – conquer scheme to convert long sequence's DFT to several iterations of short sequence's DFT, which reduce the computational complexity of DFT from $O(N^2)$ to $O(N\log N)$[1]. The length of the short sequence's DFT is called the radix of Cooley – Tukey algorithm. Large radix algorithm means less iterations, higher data throughput and without saying, more hardware consumption.

According to Cooley – Tukey algorithms with different radix and various architectures, many FFT processors have been proposed, as shown in table 1. Radix – 2^x Single – path Delay Feedback (SDF) architecture made a good compromise between speed, flexibility, hardware consumption and control complexity, as shown in figure 5[8,9], in which BFI and BFII represent two different types of butterfly units.

However, when $N > R$, which means resolution in frequency domain will increase, the sliced subsequences should be augmented with $N - R$ zero – value samples before performing N – point DFT. Large amount of zero value samples result in wasting of arithmetic operations and moreover, hardware resources. For example, with $N = 2R$, the second half samples of the sequence are zero – value. With BFI in figure 5 performing

$$\begin{cases} X(2l) = \sum_{n=0}^{N/2-1} (x(n) + x(n+N/2)) W_{N/2}^{n} \\ X(2l+1) = \sum_{n=0}^{N/2-1} (x(n) + x(n+N/2)) W_{N}^{n} W_{N/2}^{n} \end{cases} \quad (5)$$

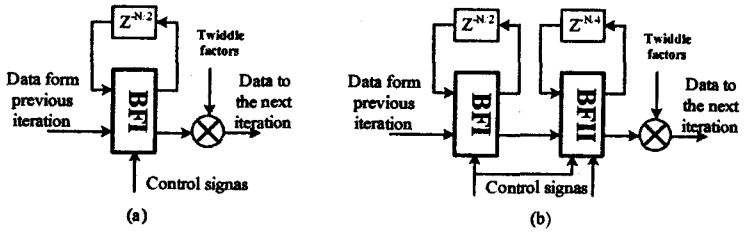

Figure 5　Iteration stage of Radix – 2^x SDF architecture with(a)x=1,(b)x=2

addition and subtraction operations are totally wasted, which means the first stage of butterfly unit should be omitted. An optimized Radix – 2^3 – 2^2 architecture for N

Table 1　Comparison of Hardware Requirement and Power Consumption

Architectures		Complex multiplier number	Complex adder Number	Memory size	Complex Multiplication per FFT	Complex Addition per FFT	Memory access per FFT
Pipeline	R4SDC[10]	$\log_4 N-1$	$8\log_4 N$	$2N-2$	$3/4N(\log_4 N-1)$	$N\log_2 N$	$2N\log_4 N$
	R2SDF[9]	$\log_2 N-2$	$2\log_2 N$	$N-1$	$1/2N(\log_2 N-2)$	$N\log_2 N$	$2N\log_2 N$
	R4SDF[11]	$\log_4 N-1$	$8\log_4 N$	$N-1$	$3/4N(\log_4 N-1)$	$N\log_2 N$	$6N\log_4 N$
	R2²SDF[8]	$\log_4 N-1$	$4\log_4 N$	$N-1$	$3/4N(\log_4 N-1)$	$N\log_2 N$	$2N\log_2 N$
	R2MDC	$\log_2 N-2$	$2\log_2 N$	$1.5N-2$	$1/2N(\log_2 N-2)$	$N\log_2 N$	$2N\log_2 N$
	R4MDC[12]	$3(\log_4 N-1)$	$8\log_4 N$	$2.5N-4$	$3/4N(\log_4 N-1)$	$N\log_2 N$	$3N\log_4 N$
	R8MDC[13]	$7(\log_8 N-1)$	$(24+2T)\log_8 N$ *	$3N-8$	$7/8N(\log_8 N-1)$	$(24+2T)/8N\log_8 N$	$7/2N\log_8 N$
Memory Based	Dual – port Memory[14]	At least 1	At least 2	N	$1/2N(\log_2 N-2)$	$N\log_2 N$	$2N\log_2 N$
	Ping – pong Memory[15]	At least 1	At least 2	$2N$	$1/2N(\log_2 N-2)$	$N\log_2 N$	$2N\log_2 N$

* T is the corresponding additions number for multiplication with a complex constant.

$= 2R = 32$ is shown in figure 6, one of the complex multipliers is replaced by a constant multiplier in comparison with traditional Radix -2^2 SDF architecture proposed in reference[8]. The first stage of butterfly unit is replaced by a multiplexer controlled by the MSB of the counter is proposed as shown in figure 7.

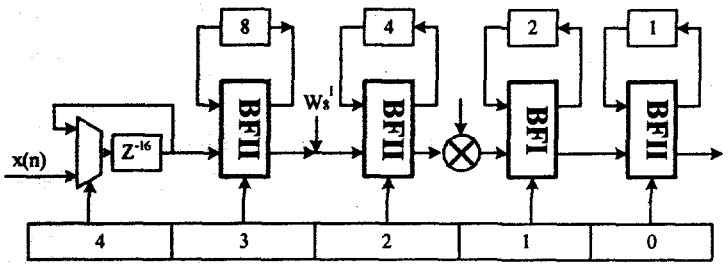

Figure 6 Optimized Radix -2 SDF for $N = 2, R = 32$

Generally, if $N = 2^rR$, the first r stages of butterfly units can be replaced by multiplexers. The feedback FIFO of the ith stage can be set to

$$\begin{cases} D(i) = N/2^r, i < r \\ D(i) = N/2^i, i \geqslant r \end{cases} \tag{6}$$

instead of $\frac{N}{2^i}$. Since multiplication operation can be bypassed when multiplied with W^0, the utilization rate of a multiplier can be defined as the proportion of non $-$ one value twiddle factors out of the total. Therefore, the utilization rate of the multiplier cascading the ith stage butterfly is

$$\begin{cases} U_{mul}(i) = \left(1 - \frac{1}{2^x}\right) \cdot 2^{i-r}, i < r \\ U_{mul}(i) = 1 - \frac{1}{2^x}, \qquad i \geqslant r \end{cases} \tag{7}$$

in which 2^x is the radix of Cooley $-$ Tukey algorithm. The same definition for butterfly utilization rate can be achieved with value of 50%. When the multipliers/butterflies have low utilization rate, operations performed on them can be combined and performed by less hardware. The lower the utilization rate is, the more probability of removing them there would be.

3 Discrete STFT Processor Architecture

Because SDF FFT unit needs at least N cycles to perform N $-$ point FFT for each subsequence, decimation for L means each N $-$ point FFT should be accomplished every L sample cycles. Therefore, the STFT processor needs $\lceil N/L \rceil$ SDF FFT units totally for real $-$ time time $-$ frequency analysis demand.

The first consideration is to parallelize all the $\lceil N/L \rceil = \lceil 32/3 \rceil = 11$ FFT units as shown in figure 7. Input data come from each node of the shift register, multiply with window factors and stream into the separated FFT units in the dashed boxes. In figure 8, BFI and BFII are both represented as BF for simplicity. Each SDF FFT unit performs FFT for consecutive segments, which means the overlapped segments' FFTs are performed simultaneously by different SDF FFT units.

A simple example for $N = R = 4$, $L = 1$ discrete STFT calculation process is shown in figure 8, in which FFTs for each windowed segment is performed in dashed boxes. In this architecture, each FFT unit starts to perform FFT at the same time, thus all the 11 FFT units can share the uniform controller. Furthermore, window factors streamed to all the N window factor multipliers are identical in a certain cycle, which means only one access of window factor memory is required per clock cycle. The same situation appears at the twiddle factor memory access in each SDF FFT unit.

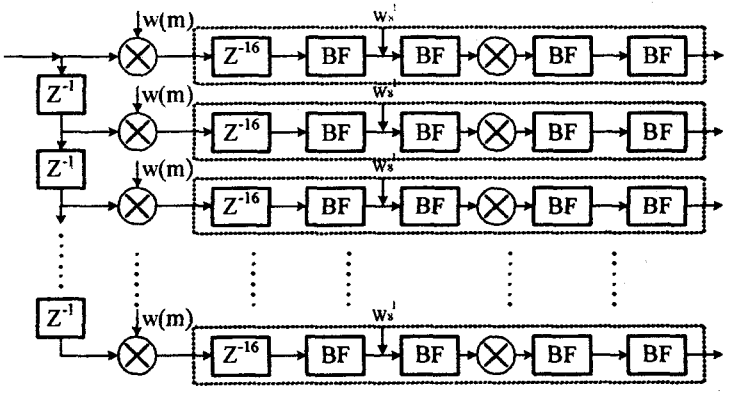

Figure7 Original design of the discrete STFT

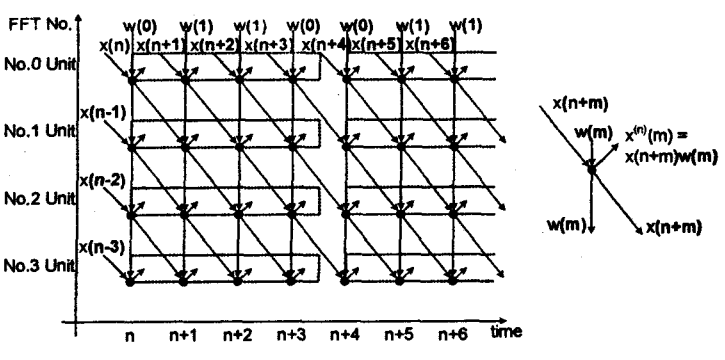

Figure8 Discrete STFT calculation process

However, as shown in figure 8, symmetry of window factors is not taken into account totally. For the nth SDF FFT unit, the input data is

$$x^{(n)}[m] = x[n+m]w[m]$$
$$= x[(n-N+2m+1)+(N-1-m)]w[N-1-m]$$
$$= x^{((n-N+2m+1)\%N)}[N-1-m] \quad (8)$$

It is also the input data of some other SDF FFT unit at some other time. Generally, because the window function slides with the overlapped segments, each input data will multiply with all the N window factors once throughout the calculation process. And with the symmetry of window factors, $N/2$ multiplication per sample can fulfill the window multiplication operation, while N multiplications are performed per sample in figure 7.

Furthermore, according to equation (7), it can be shown that some multipliers' utilization rate in FFT units is much less than 100%. However, because the parallel multipliers are either idle or on duty simultaneously, multipliers can not be reduced even their utilization rate is 50%, as the situation at constant multiplication with W_8^1. Moreover, this situation causes another problem in power consumption. Even if the multipliers can be bypassed when multiplied with 1 and average power consumption is reduced, peak power remains steady. When peak power consumption is achieved, the worst case, which is more pessimistic than the average power consumption, must be fulfilled during power supply design.

In order to develop the symmetry of the window factors, as well as remove the simultaneity of multipliers in FFT units, calculation process can be changed as shown in figure 9, and an improved architecture based on it is proposed in figure 10. Delay units between each FFT units are removed, and input data broadcast to N multipliers for window multiplication. Once the identical window multiplications take place at some time, one of the multipliers with identical operands can be bypassed to save power. Moreover, because the N FFT units start performing FFT successively, twiddle factor multipliers are on duty alternately. Therefore, twiddle factor multipliers with low utilization rate can be combined by cascading switches, thus resulting in decreased multiplier number.

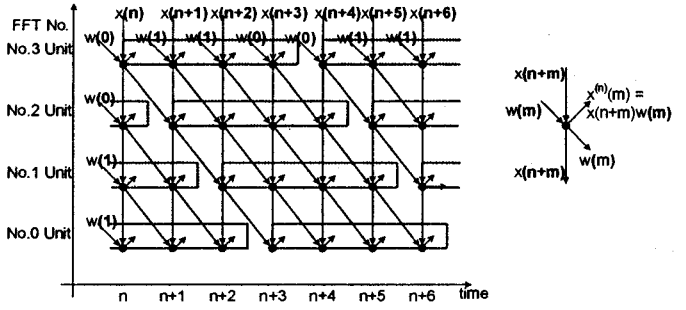

Figure9 An alternative calculation process

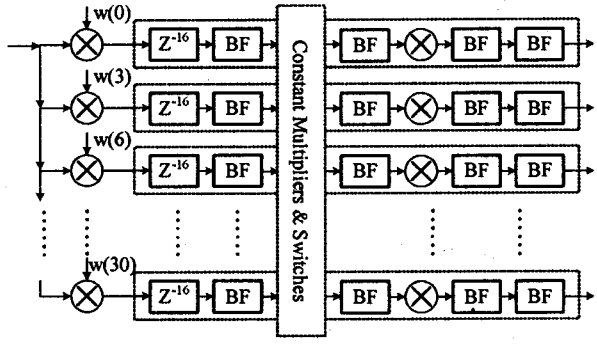

Figure10 An improved architecture

In SDF architecture, the utilization rate of all the butterfly units is 50% as mentioned above. Therefore, two butterfly units can be combined as in figure 11 providing $s^{(i)} = \overline{s^{(j)}}$ all over the time. Therefore, many of them can be omitted in the parallel FFT array.

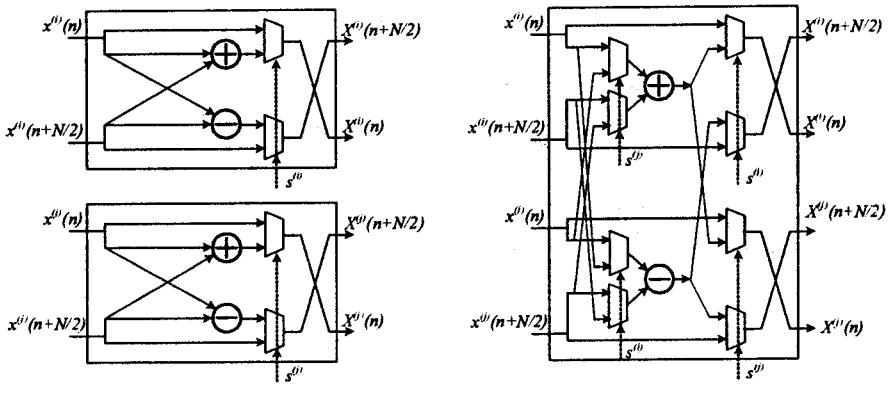

Figure11 Combination of butterfly units

4 Implementation Results

A design for $N = 2R = 32$ and $L = 3$ based on figure 11 is synthesized using SMIC 0.18μm standard cell library. Because shift registers in the design are small in

depth, it was implemented using DFFs. Appling the optimized high frequency resolution Radix -2^3-2^2 SDF architecture instead of traditional one proposed in reference [8], 11 complex multipliers converts to 11 constant multipliers for W_8^1, and 11 butterflies are saved in the STFT processor. With STFT architecture changed from figure 7 to figure 10, redundant window multiplication can be bypassed, 17 pairs of butterflies can be combined, and half of the constant multipliers for W_8^1 can be removed. In this paper, only the multipliers with 50% utilization rate are combined for switch array simplicity. Hardware and power consumption of the two of STFT processors are listed in table 2.

Table 2 Comparison between STFT processors

STFT processor	Cell area(mm^2)	Power(mw)
Before optimized	6.17	1.45
After optimized	4.73	1.16

5 Conclusion

In this paper, drawbacks of previous discrete STFT architecture are addressed. To overcome these drawbacks and make the discrete STFT processor more universal, a new architecture is proposed. Some optimizations have been done considering the hardware requirements and power consumption. A design for $N = 2R = 32$ and $L = 3$ has been synthesized using Verilog model, using SMIC 0.18μm standard cell library. The synthesized design consists of 470k equivalent logic gate, which is about 32% smaller than the traditional implementation, and the power consumption reduces by 20%. After placed and routed, static timing analysis and simulation with timing back annotated shows that under worst-case conditions, 200MHz clock rate can be achieved, which means that the real-time time-frequency analysis demand can be fulfilled under the same sample rate.

References:

[1] Cooley J W, Tukey J W. An Algorithm for Machine Computation of Complex Fourier Series[J]. Math Comput, 1965,19(4):297-301.

[2] Francos A, Porat M. Non-stationary signal processing using time-frequency filter banks[C]// Proceedings of the 13th International Conference on Digital Signal Processing. [s. l.]: IEEE, 1997:765-768.

[3] Cao Fan, Wang Shuxun, Wang Fei. A New Time-Frequency Analysis Method of Multi-Component Chirp Signal[C]// Proceedings of the! 8th International Conference on Signal Processing. [s. l.]: IEEE, 2006.

[4] Nawab S, Quatieri T, Lim J. Signal Reconstruction from Short-Time Fourier Transform Magnitude[J]. IEEE Transactions on Acoustics, Speech, and Signal Processing, 1983,31(4):986-998.

[5] Goertzel G. An Algorithm for Evaluation of Finite Trigonometric Series[J]. American Mathematical Monthly, 1958,65:34-35.

[6] Farhang-Boroujeny B, Lim Y C. A Comment on Computational Complexity of Sliding FFT[J]. IEEE Trans on Circuits and Systems II, 1992,39(12):875-876.

[7] Jacobsen E, Lyons R. The Sliding DFT[J]. IEEE Signal Processing Mag, 2003, 20(2):74-80.

[8] He Shousheng, Torkelson M. A new approach to pipeline FFT processor[C]// Proceedings of the 10th International Parallel Processing Symposium. [s. l.]: IEEE, 1996:766-770.

[9] Wold E H, Despain A M. Pipeline and parallel-pipeline FFT processors for VLSI implementation[J]. IEEE Trans Computers, 1984,C-33(5):414-426.

[10] Bi G, Jones E V. A pipelined FFT processor for word sequential data[J]. IEEE Trans Acoust, Speech, and Signal Processing, 1989,37(12):1982-1985.

[11] Despain A M. Fourier transform computer using CORDIC iterations[J]. IEEE Trans Computers, 1974,C-23(10):993-1001.

[12] Bidet E, Castelain D, Joanblanq C, et al. A fast single-chip implementation of 8192 complex point FFT[J]. IEEE J Solid-State Circuit, 1995,30(3):300-305.

[13] Zhang Shiqun, Yu Dunshan, Sheng Shimin. An ultra-high speed Real-time FFT processor[C]// Proceedings of the 6th International Conference on ASIC. [s. l.]: IEEE, 2005:227-230.

[14] Lo Hsin-Fu, Shieh Ming-Der, Wu Chien-Ming. Design of an efficient FFT Processor for DAB system[C]// Proceedings of 2001 Symposium on Circuits and Systems. [s. l.]: IEEE, 2001:654-657.

[15] Bass B M. A low-power, high-performance, 1024-point FFT Processor[J]. IEEE Jour Solid-State Circuit, 1999,34(3):380-387.