SCIENCE CHINA
Information Sciences

• **Supplementary File** •

A Novel Word length Optimization Method for Radix-2^k Fixed-Point FFT

Chen YANG[1], Yizhuang XIE[1] & He CHEN[1]*

[1]*Beijing Key Laboratory of Embedded Real-time Information Processing Technology, Beijing Institute of Technology, Beijing* 100081*, China*

Appendix A Non-trivial twiddle factors for radix-2^k FFT algorithm

The idea of radix-2^k algorithms is to try to achieve both a simple butterfly and a reduced number of twiddle factor multiplications at the same time. As the order k increases, more twiddle factors are replaced by constant factors. The essential difference between the radix-2^k algorithms is the distribution of the twiddle factors. Table A1 shows the non-trivial twiddle factor number n_i in stage i in radix-2^k algorithms.

Table A1 Number of non-trivial twiddle factors in radix-2^k algorithm

Algorithm	$n_i(i=1,2,...,\log_2 N)$	
Radix-2	$N/2^i-1$	
Radix-2^2	$\left(N/4^{i/2}-1\right)\times 3\times 4^{i/2-1}$	$\bmod(i,2)=0$
	0	$\bmod(i,2)=1$
Radix-2^3	$\left(N/8^{i/3}-1\right)\times 7\times 8^{i/3-1}$	$\bmod(i,3)=0$
	0	$\bmod(i,3)=1$
	$N/4$	$\bmod(i,3)=2$
Radix-2^4	$\left(N/16^{i/4}-1\right)\times 15\times 16^{i/4-1}$	$\bmod(i,4)=0$
	0	$\bmod(i,4)=1$
	$N/4$	$\bmod(i,4)=2$
	$3N/4$	$\bmod(i,4)=3$

Appendix B Derivation of the word length scaling variable T_i

In order to make it feasible to derive T_i, we perform an approximation as follows:

$$SQNR \approx P_X/(P_{E\_ini}+P_A). \tag{B1}$$

Define that:

$$SQNR_0=\begin{cases}12\cdot\sigma_x^2/2^{-2b_0} & rounding\\ 3\cdot\sigma_x^2/2^{-2b_0} & truncation\end{cases}, \tag{B2}$$

$$A_i=\alpha_i\cdot 2^{-3i}, \tag{B3}$$

$$B=(1/4)^{\sum_{i=1}^{v}T_i}, \tag{B4}$$

* Corresponding author (email: chenhe@bit.edu.cn)

$$C_i = (1/4)^{\sum_{j=i+1}^{v} T_j - \sum_{k=1}^{i} T_k}. \tag{B5}$$

Then B1 is expressed as follows:

$$SQNR = \frac{B}{\sum_{i=1}^{v} [C_i \cdot A_i] + 1} \cdot SQNR_0. \tag{B6}$$

Define that:

$$Q = (1/4)^{-\sum_{i=1}^{v-1} T_i}, \tag{B7}$$

$$K_i = (1/4)^{\sum_{j=i+1}^{v-1} T_j - \sum_{k=1}^{i} T_k}, \tag{B8}$$

$$P = \sum_{i=1}^{v-1} K_i A_i, \tag{B9}$$

$$R = SQNR_0/SQNR, \tag{B10}$$

$$x = (1/4)^{-T_v}. \tag{B11}$$

Then B6 is induced as follows and x is the root of the equation:

$$x^2 + \frac{1}{Q \cdot A_v} \cdot x + \frac{P}{Q \cdot A_v} - \frac{R}{Q^2 \cdot A} = 0. \tag{B12}$$

Finally the expression of T_i is derived as follows:

$$T_i = \begin{cases} \frac{1}{2}\log_2\left(\frac{R}{Q} - P\right) & \alpha_i = 0 \\ \frac{1}{2}\log_2\left(\frac{-1+\sqrt{1-4A_i \cdot (Q \cdot P - R)}}{2A_i \cdot Q}\right) & \alpha_i \neq 0 \end{cases}. \tag{B13}$$

For the reason that x must be a positive number, the negative root is rejected.

Appendix C Memory bit count calculation for fixed-point pipeline FFT

Single feedback delay (SDF) architecture is one of the most frequently-used pipeline architectures. A 256-point SDF radix-2^2 FFT is shown in Figure C1. $b1 \sim b8$ are the bit widths of the feedback memories ($S1 \sim S8$).

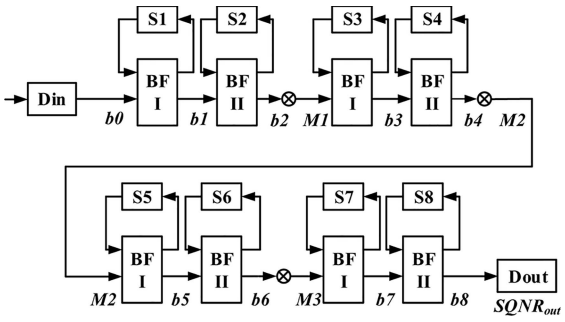

Figure C1 256-point Radix-2^2 SDF pipeline FFT architecture.

Accordingly, the total memory bits amount M_T of a N-point DIF SDF complex FFT is expressed as follows:

$$M_T = 2 \cdot \sum_{i=1}^{v} \frac{b_i}{2^i} \cdot N. \tag{C1}$$