New Algorithms for Computing a Single Component of the Discrete Fourier Transform

G. Jerônimo da Silva Jr., R. M. Campello de Souza and H. M. de Oliveira  
Dept. of Electronics & Systems  
UFPE, CP7800, 50711-970 Recife PE, Brasil  
E-mail: gilsonjr@gmail.com.

Abstract

This paper introduces the theory and hardware implementation of two new algorithms for computing a single component of the discrete Fourier transform. In terms of multiplicative complexity, both algorithms are more efficient, in general, than the well known Goertzel Algorithm.

1 Introduction

Discrete transforms are mathematical tools used in many applications in Engineering. A particularly significant example is the discrete Fourier transform (DFT) [1]. Let $v = (v_n)$, $n = 0, \ldots, N-1$, be a sequence of complex numbers or of real numbers. The DFT of $v$ is the sequence of complex numbers $V = (V_k)$, $k = 0, \ldots, N-1$, defined by

$$V_k \triangleq \sum_{n=0}^{N-1} v_n W_N^{kn},$$

where $W_N = e^{-j\frac{2\pi}{N}}$ and $j = \sqrt{-1}$.

The polynomial representation for an input signal $v$, denoted by $v(x)$, is defined by

$$v(x) \triangleq \sum_{n=0}^{N-1} v_n x^n.$$  

(2)

Therefore, the component $V_k$ can be computed from $v(x)$ by

$$V_k = v(W_N^k).$$

(3)

From [1], the computation of a single coefficient $V_k \in \mathbb{C}$, requires $N-1$ complex multiplications, $N-1$ complex additions and the prestorage of the coefficients $W_N^k$. An algorithm to implement this computation, without the need for storing the coefficients, was presented in [2]. The Goertzel algorithm, as it became known, computes the component $V_k$ via the polynomial

$$p_k(x) = (x - W_N^k)(x - W_N^{-k})$$

$$= 1 - 2 \cos \left( \frac{2\pi k}{N} \right) x + x^2,$$

which is the minimal polynomial of $W_N^k$ over the field of real numbers. It is possible to write $v(x)$ as

$$v(x) = p_k(x)q(x) + r(x),$$

(5)

where $q(x)$ and $r(x)$ are obtained by polynomial division. Since $p_k(x)$ has a zero in $W_N^k$, [3] can be used to derive

$$V_k = r(W_N^k).$$

(6)

If $v$ has real coefficients, the polynomial division by $p_k(x)$ requires $N-2$ real multiplications. Two real multiplications are necessary to compute $r(W_N^k)$, so that the Goertzel algorithm requires $N$ real multiplications to compute one component of an $N$-point DFT. The polynomial division can be implemented by an autoregressive filter, as shown in Figure 1.

![Figure 1: Autoregressive filter to compute the polynomial division by $p_k(x)$](image)

Although the Goertzel algorithm can be used to compute the DFT of a given sequence, it is not a fast Fourier transform because its computational complexity, for an $N$-point DFT, is proportional to $N^2$. It is an attractive procedure for application scenarios where only a few components (not more than $\log_2 N$ of the $N$ components) of the DFT need to be computed, such as in the detection of DTMF signals [3].

Cyclotomic polynomials play an important role in the algorithms introduced in this paper. The $N$-th cyclotomic polynomial, denoted by $\Phi_N(x)$, is the
monic polynomial which has as its roots all order \( N \) elements in \( \mathbb{C} \). Therefore
\[
\Phi_N(x) = \prod_{\theta | \text{ord}(\theta) = N} (x - \theta).
\]
and
\[
\prod_{d|N} \Phi_d(x) = (x^N - 1).
\]

From the Möbius inversion formula \([4]\), it can be shown that
\[
\Phi_N(x) = \prod_{d|N} (x^d - 1)^{\mu(N/d)},
\]
where \( \mu(n) \) is the Möbius function \([5]\)
\[
\mu(n) = \begin{cases} 1, & \text{if } n = 1; \\ 0, & \text{if } \exists e_i \geq 2; \\ (-1)^m, & \text{otherwise}, \end{cases}
\]
and \( n \) has the canonical factorization \( n = p_1^{e_1} p_2^{e_2} \cdots p_m^{e_m} \). The degree of \( \Phi_N(x) \) is given by \( \phi(N) \), where \( \phi(.) \) is the Euler totient function \([5]\).

In this paper, a new algorithm for computing a single DFT component, \( V_k \), using the JCO algorithm, considers the cyclotomic polynomial \( \Phi_L(x) \), where
\[
L = \text{ord}(W_N^k) = \frac{N}{\gcd(N,k)}.
\]
Then, by definition, \( \Phi_L(x) \) has a zero in \( W_N^k \) and \( v(x) \) can be written as
\[
v(x) = \Phi_L(x)Q(x) + R(x),
\]
where \( R(x) \) can be computed by an autoregressive filter and
\[
V_k = R(W_N^k).
\]

Unlike the \( p_k(x) \) polynomial, \( \Phi_L(x) \) has integer coefficients which, for \( L \) smaller than 105, are equal to 0, 1 and \(-1\)\([6]\). Therefore no multiplication is required to compute the polynomial division. The polynomial \( \Phi_L(x) \) has degree \( \phi(L) \), so that \( 2|\phi(L) - 1 \) real multiplications are needed to compute \( V_k \) using the JCO algorithm.

### 3 The JCO-Goertzel Algorithm

The computation of \( R(x) \) from \( v(x) \) via the JCO algorithm is multiplication free. The polynomial \( R(x) \) in \([12]\) has degree \( \leq (\phi(L) - 1) \) and can be written as
\[
R(x) = p_k(x)q(x) + v(x),
\]
from which the \( V_k \) component can be computed by the Goertzel algorithm, as in \([9]\). Therefore, the number of real multiplications in the JCO-Goertzel algorithm is \( \phi(L) \). Due to the fact that
\[
\phi(L) < L \leq N,
\]
it is clear that the JCO-Goertzel algorithm is more efficient, in terms of multiplicative complexity, than the Goertzel algorithm.

Table \([1]\) shows the multiplicative complexity (real multiplications) of the Goertzel, JCO and JCO-Goertzel algorithms, for some values of \( N \) and \( k \), assuming that \( v_k \in \mathbb{R} \). When \( \phi(L) = 2 \), the cyclotomic polynomial \( \Phi_L(x) \) is equal to \( p_k(x) \) and the multiplication by the coefficient \( A = 2 \cos(2\pi k/N) \) is a trivial one. Consequently, for \( L = 3, 4, 6 \) in Table \([1]\) the algorithms present the same performance. From \([12]\), it is clear that the only case for which the Goertzel algorithm outperforms JCO is when \( L = N \) and \( N \) is a prime number, as indicated in Table \([1]\) for \( N = 83 \).

| \( N \) | \( k \) | Goertzel | JCO | JCO-Goertzel | \( L \) |
|---|---|---|---|---|---|
| 12 | 1 | 12 | 6 | 4 | 12 |
| 2 | 2 | 2 | 2 | 2 | 4 |
| 3 | 2 | 2 | 2 | 4 |
| 4 | 2 | 2 | 2 | 3 |
| 32 | 1 | 32 | 32 | 32 | 32 |
| 2 | 32 | 14 | 8 | 16 | 16 |
| 3 | 32 | 30 | 16 | 32 | 32 |
| 4 | 32 | 6 | 4 | 8 |
| 48 | 1 | 48 | 30 | 16 | 48 |
| 2 | 48 | 14 | 8 | 24 |
| 3 | 48 | 14 | 8 | 16 |
| 4 | 48 | 6 | 4 | 12 |
| 83 | 1 | 83 | 83 | 83 | 83 |
| 120 | 1 | 120 | 62 | 32 | 120 |
| 2 | 120 | 30 | 16 | 60 |
| 3 | 120 | 30 | 16 | 40 |
| 4 | 120 | 14 | 8 | 30 |

### 4 Hardware Implementation

The hardware implementation of the Goertzel algorithm can be made using the autoregressive filter
\[
H(z) = \frac{1}{1 - W_N^k z^{-1}},
\]
with input $v_n$, $n = 0, \ldots, N - 1$ and output $y_n$. The filter computes

$$V_k = y_N.$$  \hspace{1cm} (17)

To derive a hardware implementation of the JCO algorithm, $H(z)$ is written as

$$H(z) = \frac{1}{\Phi_L(z^{-1})} \prod_{\text{ord}(W_k) = L, i \neq k} \left(1 - W_N^{-i} z^{-1}\right).$$  \hspace{1cm} (18)

so that, from (7) and the $H$ algorithm, the components are all trivial. An attractive aspect of this implementation is that the $v_n$ components are fed into the shift register circuit in arrival order, thus requiring no components storage.

The desired DFT component is obtained from the filter output as $V_{128} = y_{1024}$. The corresponding hardware implementation of the JCO algorithm is shown in Figure 3. The computation of $V_{128}$ requires only 2 multiplications and 1027 additions, in contrast to 1024 multiplications and 2049 additions as required by the Goertzel algorithm.

Figure 3: Example 1 JCO hardware implementation, $a_1 = \frac{\sqrt{2}}{2}(1 + j)$ and $a_3 = \frac{\sqrt{2}}{2}(-1 + j) = -a_1^*$.  

5 Conclusions

In this paper two new algorithms for computing a single component of the discrete Fourier transform, the JCO and the JCO-Goertzel algorithms, are proposed. Both algorithms have, in general, a better performance in terms of computational complexity, when compared to the well known Goertzel algorithm, which is the standard procedure for this type of computation. In particular, the JCO-Goertzel algorithm has the lowest multiplicative complexity, as far as we know, of the algorithms that compute a single component of an $N$-point DFT.

The approach presented in this paper represents a change in paradigm with respect to the Goertzel method in the sense that, instead of using a fixed polynomial of degree 2, the cyclotomic polynomial $\Phi_L(x)$ is used. $L$ is an integer that is a function of $N$, the DFT length, and $k$, the index of the DFT component to be computed. This means that different components of the DFT will be computed with different complexities. Consequently, considering that the JCO-Goertzel algorithm requires less than $N$ multiplications for each DFT component computed, it can be used to compute an $N$-point DFT with less than $N^2$ multiplications.

Application scenarios that use the Goertzel algorithm will benefit from the techniques introduced in this paper. In the field of real numbers, for instance, the detection of DTMF signals is a typical and important application. In the finite field context, the syndrome computation in the decoding of a BCH code can be implemented by the Goertzel algorithm. Therefore, considering that a finite field version of the results presented here can...
be derived following essentially the same approach, the proposed algorithms can be used to assist the decoding of such codes.

References

[1] A.V. Oppenheim, R.W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, 2nd ed. Prentice Hall, 1999.

[2] G. Goertzel, An algorithm for the evaluation of Infinite trigonometric series, *The American Mathematical Monthly*, vol. 65, no. 1, pp. 34-35, 1958.

[3] R. Beck, A. Dempster, and I. Kale, Finite-precision Goertzel Filters used for signal tone detection, *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 48, no. 7, pp. 691-700, Jul 2001.

[4] R. J. McEliece, *Finite Fields for Computer Scientists and Engineers*. Kluwer Academic Publishers, 1987.

[5] D. M. Burton, *Elementary Number Theory*, 6th ed. McGraw-Hill, 2007.

[6] R. E. Blahut, *Fast Algorithms for Digital Signal Processing*. Addison-Wesley Publishing Company, 1984.

[7] J. Beraldin and W. Steenaart, Overflow analysis of a fixed-point implementation of the Goertzel algorithm, *Circuits and Systems, IEEE Transactions on*, vol. 36, no. 2, pp. 322-324, Feb 1989.

[8] J. F. Chicharo and M. T. Kilani, A sliding Goertzel algorithm, *Signal Processing*, vol. 52, no. 3, pp. 283-297, 1996.

[9] R. Garcia-Reguet, S. Gonzalez, M. Funes, and S. Maestri. Implementation of a novel synchronization method using sliding Goertzel DFT, *Intelligent Signal Processing, 2007. WISP 2007. IEEE International Symposium on*, pp. 1-5, Oct. 2007.

[10] R. E. Blahut, *Theory and Practice of Error-Control Codes*. Addison-Wesley Publishing Company, 1984.