A Programmable Approach to Evaluate Ramanujan Sums

Lina Zhou, Zulin Wang, Jiadong Shang, Lei Zhao
School of Electronics and Information Engineering,
Beihang University, Beijing 100191, China
lyndabuaa@gmail.com

Song Pan
Computer Science and Technology College
Wuhan University

***Abstract*—Ramanujan sums, which have been widely researched in mathematics, recently began to attract more attentions in signal processing and communications. The traditional methods to get the values of Ramanujan sums follow the definition and formulas in number theory, both of which need factorization information. As it is complex and the amount of time needed is unpredictable in hardware programming, a programmable approach base on the primitive roots of unity is proposed in this paper. Only simple arithmetic computing and cosine function is involved. Considering that each value of Ramanujan sums is an integer, the number of sample points required is no more than the period of Ramanujan sums, and quantification bits required are no more than 16 bits, as the simulation results demonstrated.**

Keywords-Ramanujan sum; programmable; primitive roots of unity

I. INTRODUCTION

Ramanujan Sums (RS) are named after Indian mathematician Srinivasa Ramanujan, who used them to prove Vinogradov's theorem that every sufficiently-large odd number is the sum of three primes in 1918 [1]. After that, scientists began to realize their importance and use RS to form convergent expressions of some arithmetical functions in number theory, such as *d(n)* the number of divisors of *n*, *σ(n)* the sum of divisors, and the von Mangoldt function [1][2]. Gadiyar and Padma expanded the domain of applications of RS from number theory to time series using *Ramanujan-Fourier transform (RFT)* [3], which linked the RS to signal analysis applications, such as low-frequency noise processing [4], Doppler spectrum estimation [5] and T-wave alternans analysis [6]. Besides, thanks to the integer property, RS have been applied in simplifying the computation of AFT [7], DFT [8] [9] and DCT [10] coefficients under certain conditions.

There are various ways using RS in signal processing, however, there are few ways to calculate RS values. To date, three methods have been published including calculating according to RS's definition, an expression of Mobius function and Euler Totient function in number theory[11][12], and a z-transform method [8].These three methods can be sorted into two kinds: recursive methods and nonrecursive methods. Recursive methods need to compare a series of numbers following some rules, such as calculating according definition, while nonrecursive methods obtain the results straightforward with few expressions, which are preferred in generally speaking. However, Mobius function and Euler Totient function need number factorization information, whose complexity is hard to estimate. Also, the definition of RS based on coprime numbers needs factorization information. The z-transform method was proposed by Samadi when he and his colleagues used RS to compute DFT coefficients of even-symmetric real-valued period signals [8]. Although it is attractive for signal practitioners, the fact that different RS with different *q* need different expressions, coupled with the fact that each closed-form expression needs hand calculation, cause the focus on the other methods to calculate RS values. In order to using RS in signal processing with hardware, a simpler and more convenient method is proposed for hardware programming.

The programmable approach follows the relationship between RS and the primitive roots of unity with a recursive calculation. Only the basic functions and cosine function are involved. With the help of the integer property of RS, the number of sample points and the number of quantification bits are both cut down to an acceptable level.

The paper is organized as follows: In Section II, we introduce the definition of RS and some properties of RS in brief. In Section III, we demonstrate those three existed calculation methods mentioned above, and then discuss the relationship between RS and the primitive roots of unity in Section IV. Section V is devoted to the programmable approach to calculating RS. We give a conclusion and some further application of RS in Section VI.

II. RAMANUJAN SUMS

Ramanujan sum is a function of two positive integer variables *q* and *n* defined by the formula:

$$c_q(n) = \sum_{\substack{p=1 \\ (p,q)=1}}^{q} \exp(2i\pi n \frac{p}{q}) \qquad (1)$$

where $(p, q)=1$ means that *p* and *q* are coprime. From RS definition, we can obtain the *q*th RS value $c_q(n)$'s closed-form expression, the first 5 are as follows:

$$c_1(n) = \exp(2in\pi),$$
$$c_2(n) = \exp(in\pi),$$
$$c_3(n) = \exp(\frac{2}{3}in\pi) + \exp(\frac{4}{3}in\pi),$$
$$c_4(n) = \exp(\frac{2}{4}in\pi) + \exp(\frac{6}{4}in\pi),$$
$$c_5(n) = \exp(\frac{2}{5}in\pi) + \exp(\frac{4}{5}in\pi) + \exp(\frac{6}{5}in\pi) + \exp(\frac{8}{5}in\pi).$$

There are a number of reasons why RS are chosen as a useful tool in signal processing, partly because of the RS properties and partly because of their characteristic. Four properties are central to almost every application, and briefly they are as follows;

Property 1: integer property $c_q(n) \in Z$

Property 2: periodic property $c_q(n) = c_q(n \bmod q)$

Property 3: multiplicative property

$c_{qq'}(n) = c_q(n)c_{q'}(n), (q,q') = 1$

Property 4: orthogonality property

$$\begin{cases} \sum_{n=1}^{qq'} c_q(n)c_{q'}(n) = 0, q \neq q' \\ \sum_{n=1}^{q^2} c_q^2(n) = q^2\varphi(q), q = q' \end{cases}$$

The integer property provides signal processing operations without plural operations; the periodic property reduces the number of RS values $c_q(n)$ to no more than q for any n with a fixed q. The multiplicative property gives a way to calculate big RS values from other small RS values, and the orthogonality property is useful to find orthoganality frequencies of the signal. The reader is referred to [13] to find the proof processes of those properties. Besides, the spectrum of RS is non-uniform distributed, which helps a lot in finding some arithmetical features hidden in the signal. The spectrum of first 10 series of RS is shown in Fig.1, where the heights vary with the different q, and the x axis denotes the different normalized frequency as π*rad/sample*. Considering the symmetry of the spectrum, only the frequencies between $[0, \pi]$ are included in this figure.

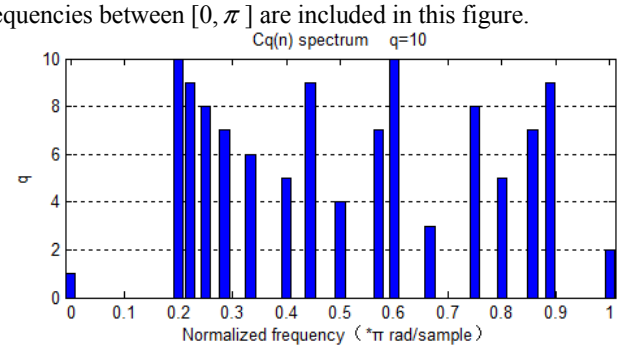

Figure 1. The spectrum of first 10 series of RS

III. Traditional Methods

Besides following RS's definition, a non-recursive method using Mobius function and Euler Totient function is widely used. RS values are evaluated from

$$c_q(n) = \mu\left(\frac{q}{(q,n)}\right)\frac{\varphi(q)}{\varphi\left(\frac{q}{(q,n)}\right)} \quad (2)$$

where $\varphi(q)$ is Euler function and $\mu(n)$ is Mobius function. These two functions are explained later. First, let (q, n) represent the greatest common divider of q and n. Factoring a number into prime numbers, q and n are transcript as:

$$q = \prod_i q_i^{\alpha_i} \quad (q_i \text{ is a } prime)$$
$$n = \prod_k n_k^{\beta_k} \quad (n_k \text{ is a } prime) \quad (3)$$

Euler's Totient function $\varphi(q)$ counts the number of positive integers less than or equal to q that are coprime to q. A commonly used computation formula is:

$$\varphi(q) = q\prod_i \left(1 - \frac{1}{q_i}\right) \quad (4)$$

We can view the Mobius function $\mu(n)$ as a coding of prime numbers, which is defined as

$$\mu(n) = \begin{cases} 0, & \text{if } n \text{ contains a square } \beta_\kappa > 1, \\ 1, & \text{if } n = 1, \\ (-1)^k, & \text{if } n \text{ is the product of } k \text{ distinct primes.} \end{cases} \quad (5)$$

The proof of (2) can also be found in [13]. It can be seen that (5) is simple, but the Euler function and Mobius function require the factors of q in RS value $c_q(n)$. As we all know that it is hard to estimate the complexity of number factorization, this method is not simple enough for hardware programming applications.

The other recursive method of the computation of RS was published by Samadi and his colleagues in 2005 [8]. With the help of cyclotomic polynomial $F_q(z)$, the one-sided z-transform of RS can be written as:

$$C_q(z) = \frac{z^{1-\varphi(q)}\frac{d}{dz}F_q(z)}{F_q(z^{-1})} \quad (6)$$

Following (6), $C_{10}(z)$ is given by

$$C_{10}(z) = \frac{4 - 3z^{-1} + 2z^{-2} - z^{-3}}{1 - z^{-1} + z^{-2} - z^{-3} + z^{-4}} \quad (7)$$

Actually, different $c_q(z)$ with different q has different expressions depending on $F_q(z)$. Thus this method only suits for calculating RS values with fixed q rather than continuous q.

IV. Using Trigonometric Expression to Calculated Ramanujan Sums

Considering that the exponential can be denoted as trigonometric function with Euler formula:

$$e^{ix} = \cos x + i \sin x \quad (8)$$

Invoking RS integer property, RS can be described by cosine function and finally equal to:

$$c_q(n) = \sum_{\substack{p=1 \\ (p,q)=1}}^{q} \cos\left(2\pi n \frac{p}{q}\right) \quad (9)$$

We can use this cosine expression (9) to discuss the programmable method in the subsequent sections. To introduce this method, we follow the definitions of the roots of unity and the primitive roots of unity.

An nth root of unity, where q=1,2,3… is a positive integer, is a complex number z which satisfies the following equation:

$$z^q = 1 \tag{10}$$

An nth root of unity is primitive if there does not exist a root of unity for some smaller k:

$$z^k \neq 1 \quad (k = 1, 2, 3, \ldots, n-1) \tag{11}$$

Given the de Moivre's formula, this is valid for all real x and integer q:

$$(\cos x + i \sin x)^q = \cos qx + i \sin qx \tag{12}$$

Let $x = 2\pi/q$, we have a primitive qth root of unity and could be verified by:

$$(\cos\frac{2\pi}{q} + i\sin\frac{2\pi}{q})^q = \cos 2\pi + i\sin 2\pi = 1 \tag{13}$$

For k=1,2,... $q-1$, we then obtain this inequation:

$$(\cos\frac{2\pi}{q} + i\sin\frac{2\pi}{q})^k = \cos\frac{2k\pi}{q} + i\sin\frac{2k\pi}{q} \neq 1 \tag{14}$$

Note that all the primitive nth root of unity must be similar to $2\pi/n$. Together with the connection between RS and primitive roots of unit that RS are sums of series of primitive roots [11], a proposition is assumed:

Proposition 1: The primitive nth root of unity is composed of $2k\pi/q$ except the primitive kth root of unity, where $0<k<q$ and $q>0$.

The proof of the proposition is referred to the appendix. The first 5 primitive roots of unity are listed in Table I as an example which shows that the primitive roots are corresponding to the RS expressions from their definition mentioned in Section II.

TABLE I. LIST OF THE FIRST 5 PRIMITIVE ROOTS OF UNITY

q	***Primitive qth roots of unity***	***The number of qth roots of unity***
1	0	1
2	π	1
3	$\pm\frac{2}{3}\pi$	2
4	$\pm\frac{2}{4}\pi$	2
5	$\pm\frac{2}{5}\pi, \pm\frac{4}{5}\pi$	4

The first 5 primitive roots of unity are the same as Ramanujan sums

According to the properties of primitive roots, the number of primitive qth roots of unity is equal to $\varphi(q)$. Those primitive qth roots of unity are the coefficients of n in (1) and (9). Given P_q denotes the set of the qth primitive roots of unity, RS values can be calculated from

$$c_q(n) = \sum_{p_{qi} \in P_q} \cos(np_{qi}) \tag{15}$$

Considering that cosine function is an even-symmetric function and the primitive roots of unity are also even-symmetric, the computation of RS value could be reduced to half of the original computation.

V. ANALYSIS OF THE PROGRAMMABLE METHOD

The programmable method is based on the primitive roots of unity and taken them as the coefficients of cosine functions. The cosine function values usually store in the form of a table in FPGAs. The size of this table depends on the number of sample points in the x axis and the number of quantization bits in the y axis. To cut down the number of bits stored, we simulated the relationship between the RS value generated from the programmable approach and the number of sample points and between the number of quantization bits.

Fig.2 shows that for a prime number, the number of sample points between 0 and 2π required is near the period of RS q. However, the number of sample points needed for composite numbers is about half of the period of RS q. The relationship between the number of error RS values and quantization bits is similar to the number of sample points as shown in Fig.4 and Fig.5. When q is a prime number, the quantization bits needed is more than composite number. Generally speaking, 16 bits quantization is preferred.

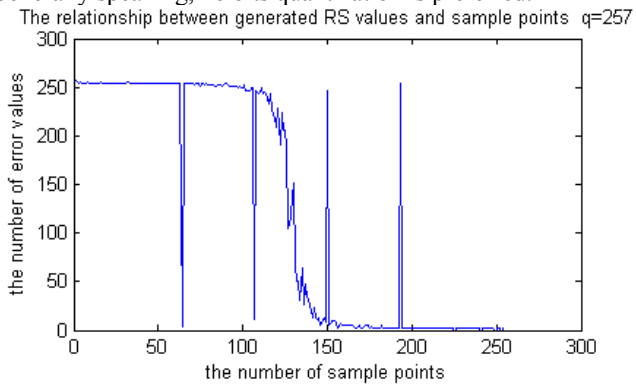

Figure 2. The relationship among RS values, sample points and prime q

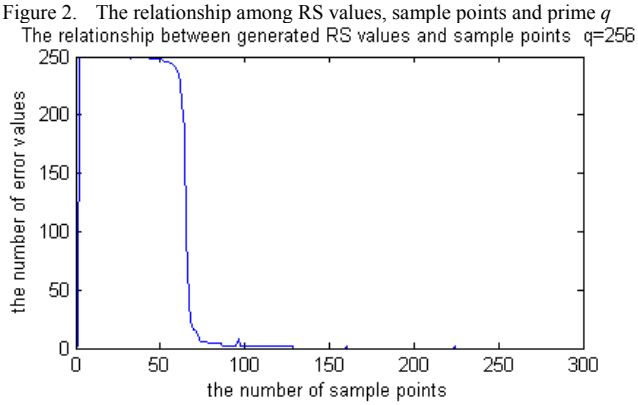

Figure 3. The relationship among RS values, sample points and composite q

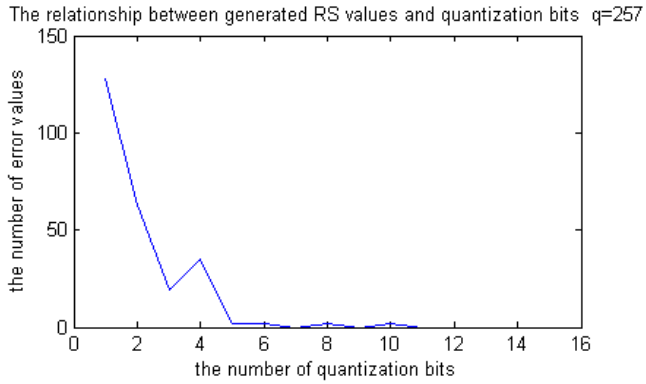

Figure 4. The relationship among RS values, sample points and prime q

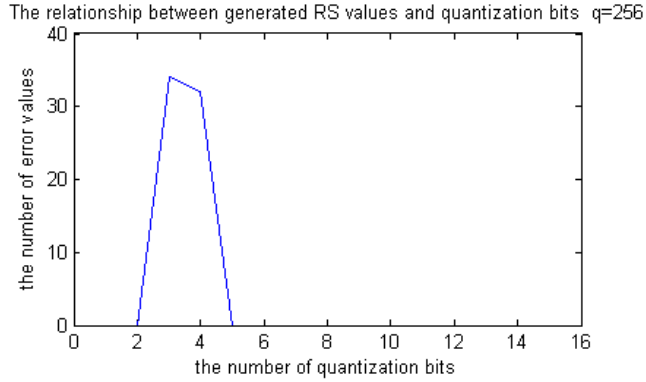

Figure 5. The relationship among RS values, sample points and compsite q

VI. Conclusion

In this article, we proposed a method based on primitive roots of unity to compute Ramanujan sums, which doesn't need to figure out number factorization, which make this method suitable for hardware programming. The required cosine function values usually store in FPGA in advance, however, the size of this bit file is small and can be used in other applications.

Furthermore, from the definition (1), $c_q(n)$ is a sum of a set of $e_p(n)$ characters, which are the basis of Discrete Fourier Transform (DFT). $e_p(n)$ are denoted by:

$$e_p(n) = \exp(2i\pi n \frac{p}{q}) \quad -1 \quad .$$

Similar with DFT, transform based on Ramanujan sums show some new advantages [4,10] and need some further research.

Acknowledgment

This work is supported by National Natural Science Foundation of China (61071070).

References

[1] S. Ramanujan, "On certain trigonometric sums and their applications in the theory of numbers," *Trans. Camb. Phil Soc.*,vol 22, pp.259-276,1918.

[2] G. hardy, proc. Cambridge philos. Soc., 20,1921,pp:263

[3] H. gadiyar, R. padma, "Linking the circle and the sieve: Ramanujan-Fourier series," *Physica A,269* ,1999, pp:503

[4] M. Planat, H. Rosu, and S. Perrine, "Ramanujan sums for signal processing of low-frequency noise," *Phys.rev.E*.vol.66 pp.56128

[5] M. Lagha, M. Bensebti, "Doppler spectrum estimation by Ramanujan-Fourier transform(RFT)," *Digital Signal Processing.2009*,19(5), pp. 843-851

[6] L. T. Mainardi, M. Bertinelli, R. Sassi, "Analysis of T-wave alternans using the Ramanujan transform," *Computer in Cardiology Bologna,* 2008,35, pp:605-608

[7] L. Knockaert, "A Generalized Mobius Transform, Arithmetic Fourier Transform and Primitive Roots," *IEEE Trans.on Signal Processing,* vol.44, no.5, 1996, pp.1307-1310.

[8] S. Samadi, M. O. Ahmad, and M. N. Swamy, "Ramanujan sums and discrete Fourier transforms," *IEEE Signal Process. Lett.*, vol. 12, no. 4, Apr. 2005, pp. 293-296 .

[9] S. C. Pei, K. W. Chang, "Odd Ramanujan Sums of Complex Roots of Unity," *IEEE Signal Processing Letters*, vol.14, no.1, 2007, pp.20-23.

[10] K. S. Geetha, V. K. Ananthashayana, "Fast multiplierless recursive transforms using Ramanujan numbers," *Proceedings of IEEE Multimedia, Signal Processing and Communication Technologies*, IEEE Press, Mar.2009, pp.116-11.

[11] G.H. Hardy,"Note on Ramanujan's trigonometical function cq(n) and certain series of arithmetical funcions",*Proc. Camb. Philip. Soc*,22,1921, pp.263

[12] E. Cohen. "An extension of Ramanujan's sum. III. Connections with totient functions," *Duke Math. J.* 23, 1956, pp:623—630

[13] P. Moree, H. Hommerson, "Value distribution of Ramanujan sums and of cyclotomic polynomial coefficients," *Mathematics Subject Classification*, 2000, pp.7-10

Appendix

Proposition 1. *The primitive nth root of unity is composed of 2kπ/q except the primitive kth root of unity, where 0<k<q and q>0.*

The proposition is proved by mathematical induction.

Proof: a) We check that the proposition is true for q=1 and q=2.

Let q=1 and $x=2k\pi/q=0/1=0$. The primitive first root of unity is $exp(0)=cos0=1$. Let q=2 and $x=2k\pi/q=2\pi/2=\pi$. The primitive second root of unity is $exp\ (ix)=cos\pi=-1$

b) Assume that $q=m-1$ and $m>2$. The proposition holds, i.e., all the primitive (m-1)th roots of unity are composed of $exp(ix)$, $x=2k\pi/(m-1)$, except the primitive kth root of unity, where $0<k<(m-1)$.

Given a natural number k, assume that

$$P_n = \{x : x = \frac{2k\pi}{q}, 0 < k < q \text{ and } \exp(i\pi x) \text{ is primitive } q\text{th root of unity}\} \tag{17}$$

denote the set of prime qth root of unity, ordered in size.

The set of all nonprimitive qth roots of m will be denoted by Q_n

$$Q_n = \{x : x = \frac{2k\pi}{q}, 0 < k < q \text{ and } \exp(i\pi x) \text{ is nonprimitive } q\text{th root of unity}\} \tag{18}$$

As we associate (17) with (18), we have:

$$U_n = P_n \cup Q_n = \{x : x = \frac{2k\pi}{q}, 0 < k < q\} \tag{19}$$

where U_n contains the 1st, 2nd,3rd...qth roots of unity.

For $q=m$, following the induction hypothesis we have U_m, P_m and Q_m.

Take an element x_j from set U_m, and we obtain

$$x_j = \frac{2j\pi}{m}, x_j \in U_m . \tag{20}$$

If for every $k = 1, 2,... ,m-1$,

$$(\cos x_j + i \sin x_j)^k \neq 1 . \tag{21}$$

thus $x_j \in P_m$.

Or there is an integer $k<m$ such that:

$$\begin{aligned}(\cos x_j + i \sin x_j)^k &= (\cos\frac{2j\pi}{m} + i\sin\frac{2j\pi}{m})^k \\ &= \cos\frac{2jk\pi}{m} + i\sin\frac{2jk\pi}{m} = 1\end{aligned} . \tag{22}$$

Then $x_j \in Q_m$, and we obtain $(j,m) \neq 1$ and

$$\frac{jk}{m} = \text{integer} . \tag{23}$$

Let *gcd(j,m)* denote the greatest common divisor of j and m,

$$\begin{aligned} j &= j' \times \gcd(j,m) \\ m &= m' \times \gcd(j,m) \\ \frac{j}{m} &= \frac{j'}{m'}, \quad m' < m \end{aligned} \tag{24}$$

From (22), (23), (24), we obtain $(j',m') = 1$, hence

$$x_j \in P_{m'}, \quad m' < m . \tag{25}$$

Notice that $Q_n = \bigcup_{0<m<n} P_m$, so the proposition holds for $q=m$.

From a) and b), the proposition statement is true.