IJCSC International Journal of Computer Science & Communication Vol. 1, No. 1, January-June 2010, pp. 73-76

Survey of Various Number Systems and Their Applications

Satrughna Singha[1] & Amitabha Sinha[2]

[1]Department of Computer Science and Engineering, JIS College of Engg., Kalyani, Inida
[2]Department of Information Technology, West Bengal University of Technology, India
Email: satrughna.singha@gmail.com

ABSTRACT

In this paper, some of the main properties of various number systems such as the Double-Base Number System (DBNS), Fermat Number System (FNS), Residue Number System (RNS), Redundant Complex Number System (RCNS) and Complex Logarithmic Number System (CLNS) have been discussed. Also, the applications of these various number systems in the area of VLSI design, digital signal processing, finite impulse response filtering etc. have been emphasized.

Keywords: DBNS, FNS, RNS, RCNS and CLNS.

1. INTRODUCTION

Binary numbers are difficult to work with because they require three or four times as many digits as their decimal equivalent. However, digital computers use binary numbers and it is sometimes necessary for the human operator or user to communicate directly with the machine by means of binary numbers. A number system using bases 2 and 3, allowing as digits only 0, 1, and requiring O(log N) nonzero digits is the DBNS which has an unusually simple 2D geometric interpretation and the binary number system is a special case and valid member of the above representation. Fermat numbers have what mathematicians sometimes describes as a "beautiful mathematical form," involving powers of 2. They were of interest 400 years ago and are now the subject of a wide-ranging worldwide computer search. In the residue number system, a number is represented by several residue digits. The arithmetic operations addition and multiplication are performed on each digit independently and allows a high degree of concurrency. An efficient structure of real / complex reconfigurable arithmetic units can be designed by the complex-number multiplier based on RCNS which exhibits highly regular structure as is observed in real-number multiplier. CLNS is very useful in areas such as FFTs, where the powers of unity have exact representation, and complex multiplication can be easily performed using fixed-point additions.

2. VARIOUS NUMBER SYSTEMS

In this section various number systems and their applications in the area of different fields have been discussed.

2.1. Double-Base Number System (DBNS)

The application of the double base number system are mainly in the area of digital signal processing and finite impulse response filtering. It also allows us for an efficient implementation of the basic arithmetic operations and considerable hardware reductions in look-up table size. Number systems are primarily chosen to enable a reduction of the complexity of the arithmetic operations as the computational complexity of algorithms crucially depends upon the number of zeros of the input data in the corresponding number system. Experimentally it has been shown that the expected number of zeros in the representation of arbitrary integers in the binary signed – digit number system shows that on average, for long word lengths, 33 percents fewer adders are needed to perform multiplication than binary. In these number systems, we need, on average, O (log N) [2] non-zero digits to represent the integer N. The DBNS uses the bases 2 and 3[1]. An extremely sparse form of the DBNS uses a single non - zero digit to represent any real number with arbitrary precision.

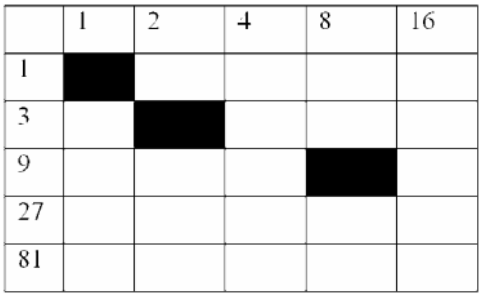

Fig 1: A 2-D representation of a DBNS (79)

The recently introduced a new double-base redundant number system with the number representation $h = \sum_i a_i.2^{b_i}.3^{t_i}$, where $a_i \in \{-1, 0, 1\}$ and

b_i, and t_i, are integers and referred to as binary and ternary exponents[1] respectively. It has been shown that the binary number system is a special case of the above representation. The DBNS has an usually simple 2-D geometric interpretation, suitable for example, for implementation via cellular automata or cellular neural networks. For example, a canonic representation of 79 is shown in figure 1.

In general the canonic form of a DBNS representation is not unique as compared to the binary canonic signed-digit redundant representation. The single digit can be mapped by its binary and ternary exponents, thus allowing an index calculus with which it can be performed arithmetic using logarithmic-like computational units. The single digit representation is shown in equation (1). $h = s2^b3^t$; where $s \in \{-1, 0, 1\}$, and b, t are signed integers. Thus h can be represented by the 3-tuple $\{s, b, t\}$. This exponent representation implies that multiplication and division are easy compared to the addition and subtraction.

2.2. Fermat Number System (FNS)

In 1640, French mathematician Pierre de Fermat (1601-1665) conjectured that all such numbers are primes, based on the observation that the first five are prime numbers. The Fermat number can be represented in two ways. The first less common way is a number of the form $2^n + 1$ obtained by setting $x = 1$ in a Fermat polynomial. The first few numbers are 3, 5, 9, 17, 33, ….[5]. The second much more commonly encountered Fermat numbers are a special case, given by the binomial number of the form $F_n = 2^{2^n} + 1$. The first few numbers for n = 0, 1, 2, …. are 3, 5, 17, 257, 65537, 4294967297, …..[5]. It can be shown that in binary notation, a Fermat number, F_n, consists of 2^{n-1} zeroes between an initial and a final 1. So, 5 is 101, 17 is 1001 and 257 is 100001. The number of digits for a Fermat number is

$$D(n) = \left\lfloor \left[\log(2^{2^n} + 1) \right] + 1 \right\rfloor \quad (1)$$

$$\cong \left\lfloor \log(2^{2^n}) + 1 \right\rfloor \quad (2)$$

$$= \left\lfloor 2^n \log 2 + 1 \right\rfloor \quad (3)$$

For n = 0, 1, …., the numbers of digits in F_n are therefore 1, 1, 2, 3, 5, 10, 20, 39, 78, 155, 309, 617, 1234, ……[5]. Euler had shown that every divisor of a Fermat number F_n with n greater than 2 has the form $k\,2^{n+1} + 1$. In the case of F_5, the divisor would be of the form 128k +1. The first prime trial divisor is 257 (k = 2), but that does not work. The second is 641 (k = 5), and it divides evenly into F_5. Thereafter, mathematician continued to look for divisors of Fermat numbers, devising a variety of methods for identifying these scarce factors[3],[4]. By 1952, a total of only 16 divisors had been identified. Being a Fermat number is the necessary (but not sufficient) form a number

$$N_n \equiv 2^n + 1 \quad (4)$$

must have in order to be prime. This can be seen by noting that if $N_n = 2^n + 1$ is to be prime, then n cannot have any odd factors b or else N_n would be a factorable number of the form

$$2^n + 1 = (2^a)^b + 1 = (2^a + 1)[\, 2^{a(b-1)} - 2^{a(b-2)} + 2^{a(b-3)} - \ldots] \quad (5)$$

Therefore, for a prime N_n, n must be a power of 2. Also, it can be shown that no two Fermat numbers have a common divisor greater than 1 [3][12]. The only known Fermat primes are

$$F_0 = 3 \quad (6)$$

$$F_1 = 5 \quad (7)$$

$$F_2 = 17 \quad (8)$$

$$F_3 = 257 \quad (9)$$

$$F_4 = 65537 \quad (10)$$

[5] and it seems unlikely that any more will be found using current computational methods and hardware. Factoring Fermat numbers is extremely difficult as a result of their large size. In fact, only F_5 to F_{11} have been complete factored. The number of factors for Fermat numbers F_n for n = 0, 1, 2, …… are 1, 1, 1, 1, 1, 2, 2, 2, 2, 3, 4, 5, ….[5].

2.3. Residue Number System (RNS)

There is no carry chain in the residue number system in which arithmetic is carried out on each digit individually. This feature is of particular interest in VLSI design. The binary number system and the decimal number system are the weighted number systems that have a carry chain. So, there is a limit on the performance of computer arithmetic. Also, the residue number system can be very useful for implementation of public key cryptography that requests the manipulation of large numbers, typically 1024 bits for most current applications. RNS[7] have the main characteristics of fast additions, fast multiplications, carry-free, high speed arithmetic, some fault detection, possible error correction and foremost parallel implementations. The residue digits are obtained by evaluating a number X modulo a set of integers m_1, m_2, …m_p, which are pairwise relatively prime, i.e., $gcd(m_i, m_j) = 1$, for $i \neq j$, $1 \leq i, j \leq p$. According to the Chinese Remainder Theorem[6], there exist a unique residue representation $(x_1, \ldots, x_p)$ for any number X in the range $[0, R-1]$, where $R = \prod_{i=1}^{p} m_i$ and $x_i = X \bmod m_i$. A number represented in residue code can be converted into binary code by the formula: $X = \left| \sum_{i=1}^{p} x_j . \widehat{m_j} . \left| 1 / \widehat{m_j} \right| \right| \bmod R$ where:

$\widehat{mj} = R / mj$ and $|\frac{1}{\widehat{mj}}|_m = a$ iff $(\widehat{mj} \cdot a) \bmod R = 1$. Let X and Y have residue codes $(x_1, \ldots, x_p)$ and $(y_1, \ldots, y_p)$ and be such that $X, Y, X + Y, X \,.\, Y \in [0, R - 1]$. Then $|X + Y|_{m_i} = |x_i + y_i|_{m_i}$ and $|X \,.\, Y|_{m_i} = |x_i \,.\, y_i|_{m_i}$ and it follows that $(|X + Y|_{m_1}, \ldots, |X + Y|_{m_p}) = (|x_1 + y_1|_{m_1}, \ldots, |x_p + y_p|_{m_p}$ and $(|X \,.\, Y|_{m_1}, \ldots, |X \,.\, Y|_{m_p}) = (|x_1 \,.\, y_1|_{m_1}, \ldots, |x_p \,.\, y_p|_{m_p})$. The higher degree of concurrency is achieved only if considering the smaller range of each digit. Minimizing the maximum modulus for a given range R maximizes the concurrency. In table 1, M is the number of bits required for a binary encoding of a digit in the residue representation of a number. N is the number of bits required for a direct binary encoding of the same number. All moduli are of the form p^{-k}, where p is prime. Moduli are listed in order of increasing p[6].

Table 1
Maximum Number of Bits in Residue Code (M) and Bits in Binary Code (N).

M	*Moduli*	*N*	*log N / M*
3	4, 3, 5, 7	8.71	1.04
4	15, 9, 5, 7, 11, 13	18.46	1.05
5	16, 27, 25, 7, 11, 13, 17,19, 23, 29, 31	46.04	1.10
6	64, 27, 25, 49, 11, 13, ….. , 53, 59, 61	~ 90	1.08
8	256, 243, 125, 49, …. , 239, 241, 251	~ 368	1.06

It can be proved that (log N) / M is asymptotically approaching 1, i.e., M = O(log N). The significance of this result is that instead of using N-bit arithmetic, log N-bit binary arithmetic suffice. Also, addition and multiplication are more complex than in the binary number system in that it is performed modulo the set of integers, $m_1, \ldots, m_p$.

2.4. Redundant Complex Number System (RCNS)

Various signal processing and scientific computation algorithms including complex orthogonal transformations, convolutions, correlations and filtering involve complex arithmetic computations. Efficient representation and manipulation are required for these kind of applications. In the usual representation scheme, the real part and the imaginary part of the given number are treated separately in arithmetic operations. Previously the most natural way of complex multiplication requires four real multiplications and two real additions. In a recently introduced method that involves three additions to generate pre-multiplication sums, three multiplications and two additions for final results. That is done by a high-speed complex number multiplier. A class of complex number representation, the Redundant Complex Number Systems (RCNSs) is originally based on the concept of "complex radix" in Knuth's quarter-imaginary number system. An RCNS is a radix-(r j) system with digits in $\{-\alpha, \ldots, 0, \ldots, \alpha\}$, where $r \geq 2$ and $[r^2 / 2] \leq \alpha \leq r^2 - 1$[8]. The complex radix r j allows unified complex number representation without treating real and imaginary part separately. Similar to Avizieni's Signed-digit (SD) number systems[9] the redundancy in number representation allows the carry-free additions and the binary-tree multiple-operand addition. In the case of RCNS with r = 2 and α = 3, conversion to and from standard binary number representation can be easily performed.

2.5. Complex Logarithmic Number System (CLNS)

A CLNS multiply requires only two fixed point additions compared to floating point representations, where a complex multiply requires four floating-point multiplies and two floating-point additions[10]. Consequently, if the cost of a CLNS add can be reduced below four floating-point multiplies and four floating-point additions, the total cost of a CLNS multiply-add will be less than floating-point representation. A complex valued number $X = X_R + X_I . i$ is represented in CLNS by its logarithm, $x = x_L + x_\theta . i$, such that $X = b^x$, where b is the base of the system. Both x_L and x_θ are fixed point numbers andcan be represented using 2's complement binary numbers. Given the representations $< x_L, x_\theta >$, $x = x_L + x_\theta . i$, and $< y_L, y_\theta >$ of two numbers X and Y, it is trivial to find the representation $< z_L, z_\theta >$ of $Z = X \times Y$ as $z_L = x_L + y_L$ and $z_\theta = x_\theta + y_\theta$. CLNS[11] addition is considerably more difficult. To compute the representation of $Z = X + Y$, it is necessary to compute z_L and z_θ.

$$z_L = x_L + f_L(x_L - y_L, x_\theta - y_\theta) \quad (11)$$

$$z_\theta = x_\theta + f_\theta(x_L - y_L, x_\theta - y_\theta) \quad (12)$$

The functions f_L and f_θ are implicitly defined in terms of r, as

$$r = x - y = x_L - y_L + (x_\theta - y_\theta) . i \quad (13)$$

$$f(r) = f_L(r) + f_\theta(r) . i \quad (14)$$

$$f(r) = \log_b(1 + b^r) \quad (15)$$

It is assumed that $x_L \geq y_L$ so that the argument to f_L and f_θ lies in the right hand half plane. Subtraction is accomplished by adding $\log_b(-1)$ to the appropriate operand[10].

3. CONCLUSIONS

In this paper, we have presented the theories of DBNS, FNS, RNS, RCNS and CLNS in short and trying to analyze some of the main characteristics of the above mentioned number systems. We have emphasized the sparseness of the Double-Base Number System. With the advent of computers and recently, a concerted effort to use the spare processing power of computers around the

world to test for divisors of Fermat numbers. One feature of the ongoing search is the race to set the record for the largest Fermat number known to be composite. It is found that in comparing with computer arithmetic based on the binary number system it shall be noticed that the residue array multiplier is of the same complexity as a binary array multiplier. By employing RCNSs, an efficient hardware algorithm for a real/complex reconfigurable arithmetic unit that can be installed into VLSI signal processors. It can be shown that a CLNS addition can be performed with approximately the same hardware as a high-radix CORDIC operation.

REFERENCES

[1] V. Dimitrov, G.A. Jullien, W.C. Miller, 1997, "Theory and Applications for a Double- Base Number System," *Proc. 13th IEEE Symp. on Comp. Arithmetic*, pp. 44-53.

[2] V. Dimitrov, G.A. Jullien, W.C. Miller, 1999, "Theory and Applications of the Double-Base Number System," *IEEE Trans. on Computers*, **48**, No. 10, Oct.1999, pp. 1098-1107.

[3] Peterson, Ivars. 2000. Great Computations. Science News 157 (March 4) : 152 -153. Available at http:// www.sciencenews.org/20000304/bob9.asp.

[4] Peterson, Ivars. 2003. Cracking Fermat Numbers. Science News (March 1) : **163**, No. 9. Available at http:// www.sciencenews.org.

[5] Slaone, N. J. A. Sequences A000051/M0717, A000215/ M2503, A005054, A019434, A046052, A057755, A050922, A070592, and A093179 in " The On-line Encyclopedia of Integer Sequences." http:// www.research.att.com/ ~njas/sequences/.

[6] C. Chiang and L. Johnson, " Residue Arithmetic and VLSI," Technical Report, California Institute of Technology, Pasadena, CA. 91125.

[7] H. Garner, "The Residue Number System," *IRE Tran. on Electronic Computers*, June 1959.

[8] T. Aoki, H. Amada, T. Higuchi, 1997, "Real/ Complex Reconfigurable Arithmetic Using Redundant Complex Number Systems," *Proc. 13th IEEE Symp. on Comp. Arithmetic (ARITH '97)*, pp. 200-207.

[9] J. Duprat, Y. Herreros, S. Kla, 1993, "New Redundant Representations of Complex Numbers and Vectors," *IEEE Trans. on Computers*, **42**, No. 7, July 1993, pp. 817-824.

[10] David Lewis, " Complex Logarithmic Number System Arithmetic Using High-Radix Redundant CORDIC Algorithm," Technical Report, University of Toronto, Ontario, Canada.

[11] D. Lewis, " A 114 MFLOPS Logarithmic Number System Arithmetic Unit for DSP Applications," *IEEE J. Solid-State Circuits*, Dec.1995, pp. 1547-1553.

[12] Hardy, G. H. and Wright, E.M. "An Introduction to the Theory of Numbers", 5th ed. Oxford, England: Clarendon Press, pp. 14-15 and 19, 1979.