For Prof. Rishe
97-EC

Proceedings of the
1997 International Conference on

Parallel and Distributed Systems

December 10–13, 1997
Seoul, Korea

Sponsored by
Parallel Processing Systems SIG of Korea Information Science Society
In cooperation with
IEEE Computer Society Technical Committee on Parallel Processing
IEEE Computer Society Technical Committee on Distributed Processing

Effective Computer Technology for Data Processing

O.D. Zhukov, N.D. Rishe

Moscow State University (Moscow)

Abstract

This paper shows the possibility to create effective computer technology for performance of typical mass algebraic procedures on the basis of special polinomial conversions and mixed number systems.

1. Introduction

In this paper a non-traditional computer technology for data representation and processing is presented. It is developed on the basis of classic number theory and some results of fundamental research of Russian academicians Chebyshev and Vinogradov [1].

A peculiarity of the algebraic constructions discussed is determined by using special polynomial conversions and mixed number systems. Some moments concerning this specific had been described in [2,3].

Special methods of representing the polynomials allow to obtain the product of two complex numbers in parallel mode using only two real multiplications instead of four as well as to obtain the product of two polynomials of a degree 'n' using n multiplications instead of n^2.

This technology is based on applying both a polynomial ring mapping (PRM) and an extended Galois Fields techniques which allows large dinamic range computations to be performed using massively parallel small finite ring computations.

Such computations can offer distinct advantages over computations using usual binary number system.

This technique allows direct mapping of bits of multiplexed binary-coded polinomial coefficients or numbers of theoretically any length to a set of independent rings, defined by the small relatively prime moduli with length not more than 5 bits in the case of using a special hardware.

In general the PRM is defined by a mapping which maps the problem of multiplication of two polynomials or complex numbers onto completely parallel scheme where the mapped polynomial coefficients are multiplied pairwise.

It is well-known from number theory if the polynomial $x^n + / - 1$ can be factorized in n distinct degree-one factors (FDOF) in a modular ring Z(m) there exists an isomorphic mapping a polynomials of a degree (n-1) onto $Z_m^n = Z_m * Z_m * ... * Z_m$.

Suppose P(m) is a finite structure containing the (n-1)st-order polynomials with coefficients in Z(m). Then by factorizing the polynomials $(x^n + / - 1)$ in n distinct factors as in

$$x^n + / - 1 = (x - r_0) * (x - r_1) * ... * (x - r_{n-1}), \quad (1)$$

$(r_i \in Z(m)$, i=0,1,...,n-1)

the product of two (n-1)st-order polynomials $mod(x^n + / - 1)$ in Z(m) can be computed with only multiplying of n pairs of the PRM mapped coefficients of these polynomials and no additions at all.

Such Factorizing $(x^n + 1)$ in n distinct Degree-One Factors (FDOF) is possible if and only if $n|(p(i) - 1)/2$, i=1,2,...,l, where $a|b$ reads "a divides b"; n and m are positive integers with prime decomposition of m given in terms of powers e(i) of its prime factors p(i), as

$$m = p(1)^{e(1)} * p(2)^{e(2)} * ... * p(l)^{e(l)}$$

with $n < p(i)$.

Similarly, for $(x^n - 1)$, the necessary and sufficient condition for its factorization becomes $n|(p(i) - 1)$.

The special case is the polynomial of kind $(x^n - 1)modm = 0$, where x=j, such that $j \in Z(m)$, p(i)=4k+1; p(i) is prime. In this case multiplication of two complex numbers is reduced to two real multiplications instead of four (without summations!).

2. Using the PRM-based Extended GF

Another effective approach to the procedure of a polynomial multiplication is possible on the basis using Galois Fields, $GF(p^e)$, and, in particular, on PRM-based extended GF represented below.

The extended $GF(p^e)$ are defined over irreducible polynomials, R(x), of order 'e', and elements in GF are computed as polynomial products (PPs), mod R over GF(p). Note that $R(x) = x^e - k$, where $k \in GF$

0-8186-8227-2/97 $10.00 © 1997 IEEE

instead, the $GF(p^e)$ multiplication is performed mod (x), where Q(x) is fully factorizable and of a suitably h degree, the above mentioned PRM decomposition be used.

A final reduction, mod R(x), realizes the $GF(p^e)$ oduct. Let us consider more detaily the $GF(p^e)$ mulication

$$W'(x) =< U(x) * V(x) > modR(x) =$$

$$=< W(x) > modR(x), \quad (3)$$

re R(x) is irreducible over GF(p), a degree 'd' of R(x) $R(x)$)) is equal 'e', and $d(U(x), V(x)) < e$.

As a degree 'd' of W(x) (d(W(x))) less than (2e W(x) can be embedded in a polynomial ring, mod), where $d(Q(x)) \geq (2 * e - 1)$ without requiring any uction of W(x), mod Q(x). Thus

$$(x) = U(x) * V(x) =< U(x) * V(x) > modQ(x), \quad (4)$$

ere $d(Q(x)) = E \geq (2 * e - 1)$.

Q(x) is chosen to factorize over GF(p) as follows:

$$Q(x) = q_0(x) * q_1(x) * ... * q_{n-1}(x), \quad (5)$$

ere the $q_i(x)$ factors are mutually prime, mod p 0,1,..,n-1). Denoting $< A(x) > modB(x)$ as the opion A(x) modB(x) for polynomials and using PRM niques, we get

$$W(x) = (< W(x) > modq_0(x),$$

$$< W(x) > modq_1(x), ...$$

$$..., < W(x) > modq_{n-1}(x)) \quad (6)$$

re the PPs, $< U(x) * V(x) > modq_i(x)$, are computed endently. Q(x) is FDOF over GF(p) if n = E, with (x)) = 1 for all 'i'. As 'p' is prime, there are (p-1) ally prime *degree − one* polynomials over GF(p). efore FDOF for an Q(x) exists only if $1 \leq E = (x)) \leq p-1$. For systems which satisfy this condition, oice of FDOF Q(x) is evident, Also, $E \geq 2 * e - 1$ is o avoid reduction of U(x)*V(x), mod Q(x). Thus, s prime,then the PRM implementation of $GF(p^e)$ plication is only possible when $e \leq (p-1)/2$. For ystems a choice of Q(x) is evident, where $2*e-1 \leq p-1$. The $q_i(x)$ are of the form $(x - g_i)$, where $1, 2, ..., p-1)$ and $g_i \neq g_j$ for $i \neq j$. In particulare $= x^E + / - 1$ and, more generally, $Q(x) = x^E - t$ be a practically useful form for Q(x),where $t \in$).

The second stage of the multiplication reduces by R(x) to obtain the $GF(p^e)$ product, $W'(x)$.

$$W'(x) =< W(x) > modR(x), \quad (7)$$

where $d(W(x)) < E$ and $d(W'(x)) < e$.

The complexity of this reduction depends on the form of R(x). The reduction, mod R(x), requires only E - e fixed multiplications and additions, mod p. To achieve this simplified reduction, an $R(x) = x^e - t, t \in GF(p)$, must be found, where R(x) is irreducible over GF(p).

The FDOF PRM technique can be generalized to the following case:

$$e_1 * e_2 * ... * e_L \geq e, \quad (8)$$

$$e_i \leq (p-1)/2 \quad (9)$$

for $1 \leq i \leq L$, that enables multiplication over $GP(p^e)$ for

$$1 \leq e \leq [(p-1)/2]^L. \quad (10)$$

As a simple case let us consider multiplication over $GF(5^8)$. The minimum value of L which (10) is satisfied for is three. Thus, FDOF PRM can be used to implement a $GF(5^8)$ mulptiplier,where $L \geq 3$. For instance, (8) and (9) are satisfied by choosing $e_1 = e_2 = e_3 = 2$. At the same time $R(x) = x^8 - 3$ is irreducible over GF(5). Because of that the R(x) reduction is simplified. Finally, (9) specifies a lower limit on p for which the method of this paper is feasible. When p=2 or 3 the e_i cannot be ≥ 1. Thus rendering the reduction is impossible.

There is another possibility of decomposition . The Agarwal-Cooley algorithm decomposes PP, mod $x^D - t$, into L-dimensional PP, mod $(x^{D_1} - t)$, mod $(x^{D_2} - t)$,...etc. (FDOF modulus must be of the form $x^D - t, t \in GF(p), D \geq e$, for this method to work). For the 2-dimensional case

$$x_1 = x^{e_1}; x_2 = x^{e_2}, \quad (11)$$

where $e_1 * e_2 \geq e$, and $gcd(e_1, e_2) = 1$.

The PP is decomposed, mod $x^{e_1 * e_2} - t$, into nested PP's, mod $x_2^{e_1} - t$ and $x_1^{e_2} - t$, respectively. However, if $e > (p-1)/2$, then $e_1 * e_2$ does not divide (p-1). This suggests a hybrid solution based on (8)-(10). So prime-factor techniques may be preferable for further decomposition.

3. Binary-Modular Algebra

In general, application of modular operations requires specialized hardware. This can create a number of problems. As alternative to the given approach the Binary-Modular Algebra (BMA) could be realized one on the basis of modified binary number system. This system would provide the data processing rate comparable with other approaches demanding specialized computers. Let 'e' be a positive integer and 'm' - the odd

modulus. The multi plicative inverse 'b' of 2^e in the ring of integers Z(m) and the multiplicative inverse 'a' of 'm' in the ring $Z(2^e)$ always exist. Therefore,the following two equations

$$< b * 2^e > modm = 1; < a * m > mod 2^e = 1 \qquad (12)$$

have a solution which can be found by Euclid's algorithm.

Now some positive integer $s < m$ can be projected into an extended ring $Z(m*2^e)$ using the general expression:

$$s' = s + k * m, \qquad (13)$$

where 's' is the projected integer in $Z(m*2^e)$ and $k<2^e$ is a positive integer.

Since the extended ring contains more elements than the original one, the isomorphism is established by constraining the projected integer to be multiplied by 2^e :

$$s' = s(e)^{(p)} * 2^e \qquad (14)$$

The relation between the original integer r in Z(m) and its "pseudo-image" $s(e)^{(p)}$, that also belongs to Z(m), can be easily found by multiplying both sides of (13) for b module made substituting (12) and (14):

$$s(e)^{(p)} =< s * b > modm \qquad (15)$$

The pseudo-images (PIs) share the same properties of original s for modular addition and substraction. In fact, if $x(e)^{(p)}$ and $y(e)^{(p)}$ are PIs of x and y with respect to the pair (m,e), the following equality holds:

$$< x(e)^{(p)} + y(e)^{(p)} > modm =$$

$$=< x * b + y * b > modm =$$

$$=<< x + y > modm * b > modm =$$

$$= [(< x + y > modm) modm](e)^{(p)} \qquad (16)$$

The modular product between two PIs $x(e)^{(p)}$ and $y(e)^{(p)}$ with respect to the pair (m,e) is equal to the PI of the product of original x and y with respect to the pair (m,2e):

$$< x(e)^{(p)} * y(e)^{(p)} > modm =$$

$$=< b < b < x * y > modm > modm > modm =$$

$$= [(< x * y > modm)](2e)^{(p)}. \qquad (17)$$

4. Implementation and Valuations

In the following, only 1-dimensional PRM is assessed. PRM is defined over a FDOF Q(x), mod p, where conversion to the PRM requires: two E-point number transforms, mod p, (for two numbers U(x) and V(x)), and one E-point Inverse number transform, mod p, for conversion from the PRM to a result W(x) . Also, E remainder products, mod p, and (E-e) fixed mults and adds, mod p, are required for the final reduction by R(x). Each number transform requires E^2 fixed mults and adds, mod p. For efficient number transforms, this figure approaches the equivalent of E general mults, mod p. Using this figure, the total count operation is: 4*E general mults, mod p; (E-e) fixed mults, mod p; and (E-e) adds, mod p.

For instance, an "optimal" dual-basis multiplier defined over an irreducible trinomial requires: e^2 general mults, mod p; 2*e fixed mults, mod p; and e^2 adds, mod p.

Comparing general mults only, the PRM design becomes competitive when $4 * E \leq e^2$ adds, mod p. Assuming, for example, E= 2*e- 1,this requires $e \geq 8$.

A complete description of realizing the represented algebraic consructions should include discussion of the following functional units: the PRM(n) unit which comprises an F(n) mapper, a parallel multiplier-adder and an $\tilde{F}(n)$ inverse mapper. The PRM(n) unit performs algebraic procedures in some modular ring Z(m) for which F(n) exists. Here the function F(n) transforms a mod m representing of polynomial coefficients to the PRM representing and the function $\tilde{F}(n)$ executes the inverse trasformation.

If the BMA (Binary-Modular Algebra) is used simpler implementation can be achieved with $m = 2^e / -1$, where e is an integer.

The PRM mapper can be implemented using mod(m) negator and a two-operand mod(m) adder.PRM mapping can be realized using a stage of scalers followed by $log_2 n$ stages of mod(m) adders (for instance, for $m = 2^e + / - 1$). A mod(m) negator can be implemented using e inverters, n-3 AND gates and n-1 EXCLUSIVE OR gates configred to achieve a propagation delay of where t is the propagation delay of a NAND gate of technology used. At the same time for the design of two-operand mod m adder a PLA Masking method can be used to achieve a propagation delay of 12t [4]. If the designed scaler has a propagation delay of 17t, the mapper requires $(17 + 12 * log_2 n)t$ in the case parallel mapper structure.

The PRM multiplier unit performs the pairwise multiplication of polynomial coefficients. Universal Multiplier Network (UMN) is used for multiplication mod $m = 2^e + / - 1$ it will require a delay $T_{PRM} = (7e + 12)t$ [4].

If $m = 2^e + / - 1$ and n is a power of two the inverse mapping can be implemented in a similar manner

the forward-mapping equations using negators and ders mod m, and regularity is observed again. The opagation delay is $(17 + 12 * log_2 n)t$ per coefficient.

A traditional multiplier mod $M = 2^d$ based on the N will have a propagation delay of about $T_{TM} = 7dt$ where d is number of binary digits of a computer d.

Thus, if the PRM multipliers are operating in multicative intensive environments , where the multiplicaus outnumber any other operations (including forward inverse mappings), then the expression

$$= T_{TM}/T_{PRM} = 7dt/(7e + 12)t = 7d/(7e + 12) \quad (18)$$

determine the speed advantages of the PRM multiers in comparison with traditional binary multipliers.

A computer number range m is a product of powof simple numbers p_i $(n|(p_i - 1)/2)$, then we can stitute e in Eq. (18) for a value E which is equal to mber of bits representing a maximum p_i. For practily used number ranges the value E is not more than Thus on the basis of (18) we obtain

$$g = 7d/(7E + 12) = 7d/49. \quad (19)$$

For example, if d=64(bits) then g=9.14.

Taking in account that traditional multiplying of degree-n polynomials includes n^2 mults and about adds and the PRM-based extended GF multiplying uires only n mults, we obtain on the basis of (18),(19)

$$G(n) \geq 7dn/49, \quad (20)$$

ere G(n) determines the performing time ratio of traional and extended GF polynomial multiplications. course, speeding up polynomial procedures is possionly if density of these procedures relatively a general lume of computations is large enough.

Conclusion

The possibility of creating an effective computer hnology using the PRM-based extended GF and MBA hniques was considered.

These techniques are defined by the transformwhich maps the problem of multiplication or addin of two polynomials onto completely parallel scheme, ere the mapped polynomial coefficients are multiplied dded pairwise. Thus, it allows to obtain a product wo degree 'n' polynomials using 'n' multiplications stead of n^2 (without summations!). Since pair-wise ltiplications of polynomial coefficients is executed in rallel and independently each to other there also is sibility to use distributed data processing.

The special case is the polynomial of kind $(x^2 + 1) mod m = 0$,where m meets (2) and p(i)=4k+1. In this case multiplication of two complex numbers is reduced to two real multiplications instead of four (without summations).

The extended $GF(p^e)$, are defined over irreducible polynomials, R(x), of order e, and elements in $GF(x^e)$ are computed as PP's,mod R(x) over GF(p). If, instead, the $GF(x^e)$ multiplication is performed mod Q(x), where Q(x) is factorizable and of a suitably high degree, the PRM decomposition can be used. A final reduction, mod R(x), realizes the $GF(p^e)$ product. By appropriate choice of PRM and R(x), all operations occure over GF(p), and reduction by an irregular polynomial modulus is eliminated. Solutions for large m and small p are possible if multidimensional PRM technique is used.

The conventional methods to perform $GF(p^e)$ multiplication use standard basis, normal basis, or dual basis scheme of multiplication, but all of them require more than e^2 general multiplications,mod p. In contrary, polynomial products (PP's) can be decomposed by the PRM into a small number of autonomous products, mod p, performed in parallel.

Of course, an effect of using the represented technique will be high if density of the described procedures relatively a general volume of computations is large enough.

It is well-known that $GF(p^e)$ multipliers are required in number of some important applications: cyclic convolutions as well as erro-correction, cryptographic, and multivalued logic systems. A further development of the represented methods can essentially extend a field of such applications.

The PRM-based extended $GF(p^e)$ system could be realized on the basis of the introduced modified binary number system BMA. The BMA could provide the rate of number processing comparable with other approaches requesting special computers.

Acknowledgment

Authors express their gratitude to NATO, NASA, and ARO for support of this research.

References

[1]. V.P. Vinogradov, "Number Theory", Science, Moscow, 1957.

[2]. O.D. Zhukov, "New Principles of Computer Construction", ITM i VT, AN SSSR, Moscow, 1975.

[3]. O.D. Zhukov, "Some Ways of Improving Computer Performance", VRE seria VT, Moscow, 1983.

[4]. Hwang K., "Computer Arithmetic," John Wiley and Sons, New York, 1979.

Effective Computer Technology for Data Processing

O.D. Zhukov, N.D. Rishe

Moscow State University (Moscow)

Abstract

This paper shows the possibility to create effective computer technology for performance of typical mass algebraic procedures on the basis of special polinomial conversions and mixed number systems.

1. Introduction

In this paper a non-traditional computer technology for data representation and processing is presented. It is developed on the basis of classic number theory and some results of fundamental research of Russian academicians Chebyshev and Vinogradov [1].

A peculiarity of the algebraic constructions discussed is determined by using special polynomial conversions and mixed number systems. Some moments concerning this specific had been described in [2,3].

Special methods of representing the polynomials allow to obtain the product of two complex numbers in parallel mode using only two real multiplications instead of four as well as to obtain the product of two polynomials of a degree 'n' using n multiplications instead of n^2.

This technology is based on applying both a polynomial ring mapping (PRM) and an extended Galois Fields techniques which allows large dinamic range computations to be performed using massively parallel small finite ring computations.

Such computations can offer distinct advantages over computations using usual binary number system.

This technique allows direct mapping of bits of multiplexed binary-coded polinomial coefficients or numbers of theoretically any length to a set of independent rings, defined by the small relatively prime moduli with length not more than 5 bits in the case of using a special hardware.

In general the PRM is defined by a mapping which maps the problem of multiplication of two polynomials or complex numbers onto completely parallel scheme where the mapped polynomial coefficients are multiplied pairwise.

It is well-known from number theory if the polynomial $x^n + / - 1$ can be factorized in n distinct degree-one factors (FDOF) in a modular ring Z(m) there exists an isomorphic mapping a polynomials of a degree (n-1) onto $Z_m^n = Z_m * Z_m * ... * Z_m$.

Suppose P(m) is a finite structure containing the (n-1)st-order polynomials with coefficients in Z(m). Then by factorizing the polynomials $(x^n + / - 1)$ in n distinct factors as in

$$x^n + / - 1 = (x - r_0) * (x - r_1) * ... * (x - r_{n-1}), \quad (1)$$

($r_i \in Z(m)$, i=0,1,...,n-1)
the product of two (n-1)st-order polynomials $mod(x^n + / - 1)$ in Z(m) can be computed with only multiplying of n pairs of the PRM mapped coefficients of these polynomials and no additions at all.

Such Factorizing (x^n+1) in n distinct Degree-One Factors (FDOF) is possible if and only if $n|(p(i)-1)/2$, i=1,2,...,l, where $a|b$ reads "a divides b"; n and m are positive integers with prime decomposition of m given in terms of powers e(i) of its prime factors p(i), as

$$m = p(1)^{e(1)} * p(2)^{e(2)} * ... * p(l)^{e(l)} \quad (2)$$

with $n < p(i)$.

Similarly, for $(x^n - 1)$, the necessary and sufficient condition for its factorization becomes $n|(p(i) - 1)$.

The special case is the polynomial of kind $(x^2 + 1)modm = 0$, where x=j, such that $j \in Z(m)$ and p(i)=4k+1; p(i) is prime. In this case multiplication of two complex numbers is reduced to two real multiplications instead of four (without

summations!).

2. Using the PRM-based Extended GF

Another effective approach to the procedure of a polynomial multiplication is possible on the basis of using Galois Fields, GF(p^e), and, in particular, on the PRM-based extended GF represented below.

The extended $GF(p^e)$ are defined over irreducible polynomials, R(x), of order 'e', and elements in $GF(p^e)$ are computed as polynomial products (PPs), mod R(x) over GF(p). Note that $R(x) = x^e - k$, where $k \in GF(p)$. If, instead, the $GF(p^e)$ multiplication is performed mod Q(x), where Q(x) is fully factorizable and of a suitably high degree, the above mentioned PRM decomposition can be used.

A final reduction, mod R(x), realizes the $GF(p^e)$ product. Let us consider more detaily the $GF(p^e)$ multiplication

$$W'(x) =< U(x) * V(x) > modR(x) =$$

$$=< W(x) > modR(x), \quad (3)$$

where R(x) is irreducible over GF(p), a degree 'd' of R(x) ($d(R(x))$) is equal 'e', and $d(U(x), V(x)) < e$.

As a degree 'd' of W(x) (d(W(x))) less than (2e - 1), W(x) can be embedded in a polynomial ring, mod Q(x), where $d(Q(x)) \geq (2 * e - 1)$ without requiring any reduction of W(x), mod Q(x). Thus

$$W(x) = U(x) * V(x) =< U(x) * V(x) > modQ(x), \quad (4)$$

where $d(Q(x)) = E \geq (2 * e - 1)$.

Q(x) is chosen to factorize over GF(p) as follows:

$$Q(x) = q_0(x) * q_1(x) * ... * q_{n-1}(x), \quad (5)$$

where the $q_i(x)$ factors are mutually prime, mod p (i=0,1,...,n-1). Denoting $< A(x) > modB(x)$ as the operation A(x) modB(x) for polynomials and using PRM techniques, we get

$$W(x) = (< W(x) > modq_0(x),$$
$$< W(x) > modq_1(x), ...$$
$$..., < W(x) > modq_{n-1}(x)) \quad (6)$$

where the PPs, $< U(x) * V(x) > modq_i(x)$, are computed independently. Q(x) is FDOF over GF(p) if n = E, with $d(q_i(x)) = 1$ for all 'i'. As 'p' is prime, there are (p-1) mutually prime $degree-one$ polynomials over GF(p). Therefore FDOF for an Q(x) exists only if $1 \leq E = d(Q(x)) \leq p - 1$. For systems which satisfy this condition, a choice of FDOF Q(x) is evident, Also, $E \geq 2 * e - 1$ is used to avoid reduction of U(x)*V(x), mod Q(x). Thus, if 'p' is prime,then the PRM implementation of $GF(p^e)$ multiplication is only possible when $e \leq (p-1)/2$. For such systems a choice of Q(x) is evident, where $2*e-1 \leq E \leq p-1$. The $q_i(x)$ are of the form $(x-g_i)$, where $g_i \in (1, 2, ..., p-1)$ and $g_i \neq g_j$ for $i \neq j$. In particulare $Q(x) = x^E + / - 1$ and, more generally, $Q(x) = x^E - t$ could be a practically useful form for Q(x),where $t \in GF(p)$.

The second stage of the multiplication reduces W(x) by R(x) to obtain the $GF(p^e)$ product, $W'(x)$. Then

$$W'(x) =< W(x) > modR(x), \quad (7)$$

where $d(W(x)) < E$ and $d(W'(x)) < e$.

The complexity of this reduction depends on the form of R(x). The reduction, mod R(x), requires only E - e fixed multiplications and additions, mod p. To achieve this simplified reduction, an $R(x) = x^e - t, t \in GF(p)$, must be found, where R(x) is irreducible over GF(p).

The FDOF PRM technique can be generalized to the following case:

$$e_1 * e_2 * ... * e_L \geq e, \quad (8)$$

$$e_i \leq (p-1)/2 \quad (9)$$

for $1 \leq i \leq L$, that enables multiplication over $GP(p^e)$ for

$$1 \leq e \leq [(p-1)/2]^L. \quad (10)$$

As a simple case let us consider multiplication over $GF(5^8)$. The minimum value of L which (10) is satisfied for is three. Thus, FDOF PRM can be used to implement a $GF(5^8)$ mulptiplier,where $L \geq 3$. For instance, (8) and (9) are satisfied by choosing $e_1 = e_2 = e_3 = 2$. At the same time $R(x) = x^8 - 3$ is irreducible over GF(5). Because of that the R(x) reduction is simplified. Finally, (9) specifies a lower limit on p for which the method of this paper is feasible. When p=2 or 3 the e_i cannot be ≥ 1. Thus rendering the reduction is impossible.

There is another possibility of decomposition . The Agarwal-Cooley algorithm decomposes PP, mod $x^D - t$, into L-dimensional PP, mod $(x^{D_1} - t)$, mod $(x^{D_2} - t)$,...etc. (FDOF modulus must be of the form $x^D - t$, $t \in GF(p)$, $D \geq e$,

for this method to work). For the 2-dimensional case

$$x_1 = x^{e_1}; x_2 = x^{e_2}, \tag{11}$$

where $e_1 * e_2 \geq e$, and $gcd(e_1, e_2) = 1$.

The PP is decomposed, mod $x^{e_1 * e_2} - t$, into nested PP's, mod $x_2^{e_1} - t$ and $x_1^{e_2} - t$, respectively. However, if $e > (p-1)/2$, then $e_1 * e_2$ does not divide (p-1). This suggests a hybrid solution based on (8)-(10). So prime-factor techniques may be preferable for further decomposition.

3. Binary-Modular Algebra

In general, application of modular operations requires specialized hardware. This can create a number of problems. As alternative to the given approach the Binary-Modular Algebra (BMA) could be realized one on the basis of modified binary number system. This system would provide the data processing rate comparable with other approaches demanding specialized computers. Let 'e' be a positive integer and 'm' - the odd modulus. The multi plicative inverse 'b' of 2^e in the ring of integers Z(m) and the multiplicative inverse 'a' of 'm' in the ring $Z(2^e)$ always exist. Therefore,the following two equations

$$< b * 2^e > modm = 1; < a * m > mod 2^e = 1 \tag{12}$$

have a solution which can be found by Euclid's algorithm.

Now some positive integer $s < m$ can be projected into an extended ring $Z(m * 2^e)$ using the general expression:

$$s' = s + k * m, \tag{13}$$

where 's' is the projected integer in $Z(m * 2^e)$ and $k<2^e$ is a positive integer.

Since the extended ring contains more elements than the original one, the isomorphism is established by constraining the projected integer to be multiplied by 2^e :

$$s' = s(e)^{(p)} * 2^e \tag{14}$$

The relation between the original integer r in Z(m) and its "pseudo-image" $s(e)^{(p)}$, that also belongs to Z(m), can be easily found by multiplying both sides of (13) for b module made substituting (12) and (14):

$$s(e)^{(p)} =< s * b > modm \tag{15}$$

The pseudo-images (PIs) share the same properties of original s for modular addition and substraction. In fact, if $x(e)^{(p)}$ and $y(e)^{(p)}$ are PIs of x and y with respect to the pair (m,e), the following equality holds:

$$< x(e)^{(p)} + y(e)^{(p)} > modm =$$

$$=< x * b + y * b > modm =$$

$$=<< x + y > modm * b > modm =$$

$$= [(< x + y > modm) modm](e)^{(p)} \tag{16}$$

The modular product between two PIs $x(e)^{(p)}$ and $y(e)^{(p)}$ with respect to the pair (m,e) is equal to the PI of the product of original x and y with respect to the pair (m,2e):

$$< x(e)^{(p)} * y(e)^{(p)} > modm =$$

$$=< b < b < x * y > modm > modm > modm =$$

$$= [(< x * y > modm)](2e)^{(p)}. \tag{17}$$

4. Implementation and Valuations

In the following, only 1-dimensional PRM is assessed. PRM is defined over a FDOF Q(x), mod p, where conversion to the PRM requires: two E-point number transforms, mod p, (for two numbers U(x) and V(x)), and one E-point Inverse number transform, mod p, for conversion from the PRM to a result W(x) . Also, E remainder products, mod p, and (E-e) fixed mults and adds, mod p, are required for the final reduction by R(x). Each number transform requires E^2 fixed mults and adds, mod p. For efficient number transforms, this figure approaches the equivalent of E general mults, mod p. Using this figure, the total count operation is: 4*E general mults, mod p; (E-e) fixed mults, mod p; and (E-e) adds, mod p.

For instance, an "optimal" dual-basis multiplier defined over an irreducible trinomial requires: e^2 general mults, mod p; 2*e fixed mults, mod p; and e^2 adds, mod p.

Comparing general mults only, the PRM design becomes competitive when $4 * E \leq e^2$ adds, mod p. Assuming, for example, E= 2*e- 1,this requires $e \geq 8$.

A complete description of realizing the represented algebraic consructions should include discussion of the following functional units: the PRM(n) unit which comprises an F(n) mapper, a parallel multiplier-adder, and an $\tilde{F}(n)$ inverse mapper. The PRM(n) unit performs algebraic

procedures in some modular ring Z(m) for which F(n) exists. Here the function F(n) transforms a mod m representing of polynomial coefficients to their PRM representing and the function $\tilde{F}(n)$ executes an inverse trasformation.

If the BMA (Binary-Modular Algebra) is used a simpler implementation can be achieved with $m = 2^e + / - 1$, where e is an integer.

The PRM mapper can be implemented using a mod(m) negator and a two-operand mod(m) adder.PRM mapping can be realized using a stage of scalers followed by $log_2 n$ stages of mod(m) adders (for instance, for $m = 2^e + / - 1$). A mod(m) negator can be implemented using e inverters, n-3 AND gates and n-1 EXCLUSIVE OR gates configred to achieve a propagation delay of 5t, where t is the propagation delay of a NAND gate of the technology used. At the same time for the design of a two-operand mod m adder a PLA Masking method can be used to achieve a propagation delay of 12t [4]. Then the designed scaler has a propagation delay of 17t while the mapper requires $(17 + 12 * log_2 n)t$ in the case of parallel mapper structure.

The PRM multiplier unit performs the parallel pairwise multiplication of polynomial coefficients. If a Universal Multiplier Network (UMN) is used for multiplication mod $m = 2^e + / - 1$ it will require a delay of $T_{PRM} = (7e + 12)t$ [4].

If $m = 2^e + / - 1$ and n is a power of two then the inverse mapping can be implemented in a similar manner as the forward-mapping equations using negators and adders mod m, and regularity is observed again. The propagation delay is $(17 + 12 * log_2 n)t$ per coefficient.

A traditional multiplier mod $M = 2^d$ based on the UMN will have a propagation delay of about $T_{TM} = 7dt$ [4], where d is number of binary digits of a computer word.

Thus, if the PRM multipliers are operating in multiplicative intensive environments , where the multiplications outnumber any other operations (including forward and inverse mappings), then the expression

$$g = T_{TM}/T_{PRM} = 7dt/(7e + 12)t = 7d/(7e + 12) \quad (18)$$

will determine the speed advantages of the PRM multipliers in comparison with traditional binary multipliers.

A computer number range m is a product of powers of simple numbers p_i $(n|(p_i - 1)/2)$, then we can substitute e in Eq. (18) for a value E which is equal to number of bits representing a maximum p_i. For practically used number ranges the value E is not more than 5. Thus on the basis of (18) we obtain

$$g = 7d/(7E + 12) = 7d/49. \quad (19)$$

For example, if d=64(bits) then g=9.14.

Taking in account that traditional multiplying of two degree-n polynomials includes n^2 mults and about n^2 adds and the PRM-based extended GF multiplying requires only n mults, we obtain on the basis of (18),(19)

$$G(n) \geq 7dn/49, \quad (20)$$

where G(n) determines the performing time ratio of traditional and extended GF polynomial multiplications. Of course, speeding up polynomial procedures is possible only if density of these procedures relatively a general volume of computations is large enough.

5. Conclusion

The possibility of creating an effective computer technology using the PRM-based extended GF and MBA techniques was considered.

These techniques are defined by the transforming which maps the problem of multiplication or addition of two polynomials onto completely parallel scheme, where the mapped polynomial coefficients are multiplied or added pairwise. Thus, it allows to obtain a product of two degree 'n' polynomials using 'n' multiplications instead of n^2 (without summations!). Since pairwise multiplications of polynomial coefficients is executed in parallel and independently each to other there also is possibility to use distributed data processing.

The special case is the polynomial of kind $(x^2+1) mod m = 0$,where m meets (2) and p(i)=4k+1. In this case multiplication of two complex numbers is reduced to two real multiplications instead of four (without summations).

The extended $GF(p^e)$, are defined over irreducible polynomials, R(x), of order e, and elements in $GF(x^e)$ are computed as PP's,mod R(x) over GF(p). If, instead, the $GF(x^e)$ multiplication is performed mod Q(x), where Q(x) is factorizable and of a suitably high degree, the PRM decomposition can be used. A final reduction, mod R(x), realizes the $GF(p^e)$ product. By appropriate choice of PRM and R(x), all operations occure over GF(p), and reduction by an irregular

polynomial modulus is eliminated. Solutions for large m and small p are possible if multidimensional PRM technique is used.

The conventional methods to perform $GF(p^e)$ multiplication use standard basis, normal basis, or dual basis scheme of multiplication, but all of them require more than e^2 general multiplications,mod p. In contrary, polynomial products (PP's) can be decomposed by the PRM into a small number of autonomous products, mod p, performed in parallel.

Of course, an effect of using the represented technique will be high if density of the described procedures relatively a general volume of computations is large enough.

It is well-known that $GF(p^e)$ multipliers are required in number of some important applications: cyclic convolutions as well as erro-correction, cryptographic, and multivalued logic systems. A further development of the represented methods can essentially extend a field of such applications.

The PRM-based extended $GF(p^e)$ system could be realized on the basis of the introduced modified binary number system BMA. The BMA could provide the rate of number processing comparable with other approaches requesting special computers.

Acknowledgment

Authors express their gratitude to NATO, NASA, and ARO for support of this research.

References

[1]. V.P. Vinogradov, "Number Theory", Science, Moscow, 1957.

[2]. O.D. Zhukov, "New Principles of Computer Construction", ITM i VT, AN SSSR, Moscow, 1975.

[3]. O.D. Zhukov, "Some Ways of Improving Computer Performance", VRE seria VT, Moscow, 1983.

[4]. Hwang K., "Computer Arithmetic," John Wiley and Sons, New York, 1979.