고효율 산술연산기시스템 구성에 관한 연구

박춘명*

*충주대학교 전기.전자 및 정보공학부 컴퓨터공학전공

A Study on Constructing the Highly Efficiency Arithmetic Operation Unit Systems

Chun-Myoung Park*

*Dept. of Computer Engineering, School of EEIE, Chungju National University

E-mail : cmpark@chungju.ac.kr

요 약

본 논문에서는 Galois체에 기초를 둔 고효율 산술연산기 구성에 관한 한가지 방법을 제안하였다. 제안한 연산기는 기존의 방법에 비해 좀 더 규칙적이고 확장성이 용이한 이점이 있으며, 또한, 각종 멀티미디어 하드웨어 구성시의 기본인 연산기로 적용 및 응용할 수 있다.

향 후 연구과제로는 좀 더 콤팩트하고 효과적인 산술연산 알고리즘의 도출이 필요하며, 이에 논리연산기를 접목하여 산술연산 및 논리연산을 수행하는 연산전용 프로세서의 개발이 필요하다.

ABSTRACT

This paper presents a method of constructing the highly efficiency arithmetic operation unit systems(AOUS) based on fields. The proposed AOUS is more regularity and extensibility than previous methods. Also, the proposed AOUS be able to apply basic multimedia hardware. The future research is demanded to more compact and advanced arithmetic operation algorithm.

키워드

Galois Fields, Irreducible Primitive Polynomial, Arithmetic Operation Algorithm, Processor

Ⅰ. 서 론

최근 지식기반의 정보화 시대에는 멀티미디어 및 모바일 하드웨어와 소프트웨어에 기반을 둔 여러 분야가 매우 급속도로 발전되고 있으며 향 후 그 중요성이 증가 될 것이다.

특히 멀티미디어 관련 하드웨어는 지금까지의 각종 데이터 처리보다는 훨씬 방대한 데이터 양, 최적의 데이터 압축 및 복원, 초고속 전송 등의 복합적이고 고기능의 기술이 요구되고 있다. [1-3]

본 논문에서는 이러한 부분을 해결할 수 있는 방법의 한가지로 연산기시스템 구성의 한가지 방법을 제안하였다.

Ⅱ. 수학적 배경

[중요성질1] 다음 식(1)을 인수분해하여 m차 기약다항식을 구한 후 이를 0으로 하는 한 근을 α라 할 때 식(2)와 같은 원시기약다항식을 얻을 수 있다.

$$X^{\beta}-X=X(X-1)(X_{\beta}^{\beta-2}+X^{\beta-3}+ \ldots\ldots +X+1)=0 \quad (1)$$

여기서 $\beta=P^m$, P 는 소수, m 은 양의정수

$$F(\alpha)=\sum_{i=0}^{m-1}\delta_i\alpha^i$$
$$=\delta_0+\delta_1\alpha^1+\ldots\ldots+\delta_{m-2}\alpha^{m-2}+\delta_{m-1}\alpha^{m-1} \quad (2)$$

또한, 식(2)를 벡터공간으로 표시하면 다음 식(3)과 같다.

$$\underline{F(\alpha)}=[\delta_0\delta_1\ldots\ldots\delta_{m-2}\delta_{m-1}] \quad (3)$$

여기서 δ_i(i=0,1, ... ,m-1)는 α^i의 계수이다.

[중요성질2] GF(P^m)상에서 원소생성은 식(2)를 0으로 하는 한 근 α의 멱승으로 구해지며, 원소의 개수는 P^m이다. 이를 식으로 표시하면 다음 식(4)와 같다.

$GF(P^m)=\{0,\alpha^1,\alpha^2,.....,\alpha^{\beta-2}=\alpha^{-1},\alpha^{\beta-1}=1\}$ (4)

여기서 $\beta=P^m$

[중요성질3] 역원의 존재

(1) $\Theta+(-\Theta)=0$인 가법에 대한 역원 $-\Theta$가 존재한다.($\forall\Theta\in GF(P^m)$)

(2) $\Theta\cdot(\Theta^{-1})=1$인 승법에 대한 역원 Θ^{-1}이 존재한다.($\forall\Theta\in GF(P^m)$)

[중요성질4] $(\Theta+\Psi)\beta=\Theta^\beta+\Psi^\beta=\Theta+\Psi$ ($\forall\Theta,\Psi\in GF(P^m)$)

[중요성질5] $\Theta^i\cdot\Theta^j=\Theta^{i+j(\mathrm{mod}\ \beta-1)}$ ($\beta=P^m$)

여기서 r=i+j이라 하면 i+j(mod β-1)은 r(mod β-1)이며 $0\le r\le P^m-1$이다.

이상의 수학적 성질과 그 외의 본 논문을 전개하는데 필요한 수학적 성질은 참고문헌[2,13]을 참고하였다.[4-9]

III. 연산알고리즘

3-1. 가산 및 감산 연산 알고리즘

피가산원소를 e_i, 가산원소를 e_j, 가산후원소를 e_a라 하고 이들을 벡터공간으로 표현한 것을 각각 $\underline{e}_i(a_V)$, $\underline{e}_j(b_V)$, $\underline{e}_a(A_V)$라 하면 두 원소 e_i와 e_j의 가산은 다음 식(5)과 같다.

$$e_i\oplus e_j=\underline{e}_i(a_V)\oplus\underline{e}_j(b_V)=\underline{e}_a(A_V) \quad (5)$$

여기서 i,j=0,1,.....,2^m-2,2^m-1이고, $a_V,b_V,A_V\in$ GF(2) (V=0,1,.....,m-2,m-1)이고, ⊕는 modP 합이다.

특히, P=2인 경우에는 식(5)에서 살펴 본 바와 같이 $A_V=a_V\oplus b_V$ 이다. 따라서 b_V를 가산연산시의 제어입력으로 사용하면 b_V 값에 따라 a_V 값을 그대로 유지하거나 2의 보수를 취한 값이 되고, 이는 mod2 합의 수학적 성질과 같다. 이 내용을 식으로 표시하면 식(6)과 같다.

$$A_V=\begin{cases} a_V & \text{iff } b_V=0 \\ a_V' & \text{iff } b_V\neq 0 \end{cases} \quad (6)$$

따라서, P=2인 경우의 가산알고리즘을 도출하면 다음과 같다.

[단계1] 피가산원소 e_i와 가산원소 e_j를 각각 비트 벡터공간으로 표현한 $\underline{e}_i(a_V)$와 $\underline{e}_j(b_V)$로 표시한다.

[단계2] 가산원소 $\underline{e}_j(b_V)$를 제어입력으로 사용하여 b_V의 값이 "0"이면 피가산원소 $\underline{e}_i(a_V)$의 해당 비트값을 그대로 유지하고 "1"이면 2의 보수를 취한다.

[단계3] STEP2를 행한 후의 결과가 최종 가산후의 원소 $\underline{e}_a(A_V)$가 된다.

P=2인 경우의 감산연산은 Mod2의 수학적 성질에 의해 가산연산과 같다.

3-2. 승산 연산 알고리즘

피승산원소를 e_i, 승산원소를 e_j, 승산후원소를 e_M이라 하고 이들을 벡터공간으로 표현한 것을 각각 $\underline{e}_i(a_V)$, $\underline{e}_j(b_V)$, $\underline{e}_M(M_V)$라 하면 두 원소 e_i와 e_j의 승산은 다음 식(7)과 같다.

$$e_i\otimes e_i=\underline{e}_i(a_V)\otimes\underline{e}_j(b_V)=\underline{e}_m(M_V) \quad (7)$$

한편, 피승산원소, 승산원소, 승산후원소의 기약다항식을 각각 $F(\alpha)=\sum_{i=0}^{m-1}a_i\alpha^i$, $G(\alpha)=\sum_{j=0}^{m-1}b_j\alpha^j$와 $H(\alpha)=\sum_{k=0}^{m-1}M_k\alpha^k$라 하면 식(7)은 다음 식(8)과 같이 표현 할 수 있다.

$$F(\alpha)\otimes G(\alpha)=\Big(\sum_{i=0}^{m-1}a_i\alpha^i\Big)\otimes\Big(\sum_{j=0}^{m-1}b_j\alpha^j\Big)$$
$$=\sum_{i=0}^{m-1}\Big(\sum_{j=0}^{m-1}b_j\Big)a_i\alpha^{i+j}=\sum_{i+j=0}^{2m-2}a_ib_j\alpha^{i+j} \quad (8)$$

여기서 a_i, $b_j\in$GF(2)이고 i,j=0,1,.....,m-2,m-1이다.

또한, r=i+j라 하면 식(8)은 식(9)와 같고 이는 H(α)와 같아야 한다.

$$F(\alpha)\otimes G(\alpha)=\sum_{r=0}^{2m-2}a_ib_j\alpha^r=H(\alpha)=\sum_{k=0}^{m-1}M_k\alpha^k \quad (9)$$

따라서 α^r의 r은 $m\le r1\le 2m-2$ 부분과 $0\le r2\le m-1$ 부분으로 분할 할 수 있으며 α^{r1}항을 수학적 성질로부터 α^{r2}항으로 표현하여 α^k항과 일치시킬 수가 있다. 또한, 이들 α^{r2}항들이 승산기 모듈 중 제어입력생성 모듈의 입력이 되고 이 제어입력 C_L에 의해 최종 승산후원소 $\underline{e}_m(M_V)$를 얻는다. 이를 토대로 승산 알고리즘을 도출하면 다음과 같다.

[단계1] 피승산원소 e_i와 승산원소 e_j를 각각 비트 벡터공간으로 표현한 $\underline{e}_i(a_V)$와 $\underline{e}_j(b_V)$로 표시한다.

[단계2] 식(3-4)와 같이 α^{r1}항과 α^{r2}항을 각각 구한다.

[단계3] α^{r1}항을 수학적 성질로부터 α^{r2}항으로 표현하여 제어입력 C_L을 구한다.

[단계4] STEP3에서 구한 C_L의 값이 "0"이면 해당 α^{r2}항의 비트값을 그대로 유지하고 "1"이면 2의 보수를 취한다.

[단계5] STEP4를 행한 후의 결과가 최종 승산후원소 $\underline{e}_m(M_V)$가 된다.

한편, P=2인 경우의 제어입력 C_L(L=0 ,1,.....,m-2, m-1)은 식(9)의 α^{r1}항으로부터 구할 수 있다. 즉,

$\sum_{r1=m}^{2m-2} R_{r1}\alpha^{r1}$을 α^{r2}항으로 표현해 이들을 modP 합함으로써 쉽게 구할 수 있으며 이를 식으로 표현하면 다음 식(10)과 같고 제어입력 C_L의 개수는 m개이다.

$$\sum_{r1=m}^{2m-2} R_{r1}\alpha^{r1} = \sum_{r1=m}^{2m-2} R_{r1}\left(\sum_{L=0}^{m-1}\alpha^{L}\right) \quad (10)$$

따라서 승산 연산은 α^r생성 부분과 제어입력C_L 생성 부분을 합성하여 구할 수 있으며 C_L의 값이 "0"이면 식(3-5)의 M_k 값은 R_{r2} 값을 유지하고 "1"이면 R_{r2} 값에 2의 보수를 취한 값이 되고 이를 식으로 표현하면 다음 식(11)과 같다.

$$M_k = \begin{cases} R_{r2} & \text{iff } C_L=0 \\ R_{r2}' & \text{iff } C_L=1 \end{cases} \quad (11)$$

여기서 M_k, C_L, R_{r2}, $R_{r2}' \in GF(2)$이고 k, L, r2=0,1,.....,m-2,m-1이다.

3-3. 제산 연산 알고리즘

피제산원소, 제산원소 및 제산후원소를 각각 $\underline{e}_i(a_V)$, $\underline{e}_j(b_V)$, $\underline{e}_d(D_V)$라 하면 두 원소 e_i와 e_j의 제산은 다음 식12)와 같다.

$$e_i \oplus e_i = \underline{e}_i(a_V) \oplus \underline{e}_j(b_V) = e_i \otimes e_j^{-1} = \underline{e}_i(a_V) \otimes = \underline{e}_d(D_V) \quad (12)$$

여기서 e_j^{-1}은 e_j의 역원이며 b_V^*는 b_V에 대한 연원 비트벡터공간이다.

한편, 피제산원소, 제산원소 및 제산후원소의 원시기약다항식을 각각 F(α), G(α)와 H(α)라 하면 식(12)는 다음 식(13)과 같다.

$$F(\alpha) \oplus G(\alpha) = F(\alpha) \otimes [G(\alpha)]^{-1} = H(\alpha) \quad (13)$$

여기서 $[G(\alpha)]^{-1}$은 G(α)에 대한 승법역원생성다항식(multiplicative inverse element generation polynomial)이다.

특히, P=2인 경우에는 Ⅱ장의 성질2로 부터 $[G(\alpha)]^{-1}$는 식(14)와 같다.

$$[G(\alpha)]^{-1} = \left(\sum_{j=0}^{m-1} b_j \alpha^j\right)^{K-2} (\text{where, } K=2^m) \quad (14)$$

위의 내용을 토대로 제산 알고리즘을 도출하면 다음과 같다.

[단계1] 피제산원소 e_i와 제산원소 e_j를 각각 비트벡터공간으로 표현한 $\underline{e}_i(a_V)$와 $\underline{e}_j(b_V)$로 표시한다.

[단계2] 제산원소의 원시기약다항식 G(α)에 대한 승법역원생성다항식 $[G(\alpha)]^{-1}$과 역원비트벡터공간 b_V^*로 표시된 $\underline{e}_j(b_V^*)$를 구한다.

[단계3] STEP2에서 구한 $\underline{e}_j(b_V^*)$를 승산 알고리즘의 승산원소로 한다.

[단계4] 이후 부터는 승산 알고리즘의 STEP2이후와 동일하다.

Ⅳ. 고속 연산기 구성

4-1. 배분기 모듈 D_1

피연산원소인 $\underline{e}_i(a_V)$는 가산일때는 가산기 모듈의 입력으로 승산일때는 α^r 생성 모듈의 입력으로 사용된다. 그러므로 이를 수행하기 위한 제어입력 T_1과 패스 트랜지스터 G_{D1i}(i=0,1)로 모듈 D_1의 기본 셀을 구성할 수 있다. 이를 식으로 표현하면 다음 식(15)와 같고 이에 대한 진리치표는 표1과 같다.

$$a_i = \begin{cases} y_{0i} & \text{if } T_1=0 \ : \text{가산기 모듈} \\ y_{1i} & \text{if } T_1=1 \ : \text{승산기 모듈} \end{cases} \quad (15)$$

여기서 i=0,1,.....,m-2,m-1이다.

표 1. 배분기 모듈 D1의 기본셀의 진리치표
Table 1. Truth table for basic D1 cell of distribution module D1.

a_i	T_1	G_{D10}	Y_{0i}	G_{D11}	Y_{1i}
a_i	0	ON	a_i	OFF	—
a_i	1	OFF	—	ON	a_i

where, — means nonpass

4-2. 배분기 모듈 D_2

연산원소인 $\underline{e}_j(b_V)$는 가산과 감산일때는 가산기 모듈의 입력으로, 승산일때는 α^r 생성 모듈의 입력으로, 제산일때는 승법역원생성 모듈의 입력으로 사용된다. 이를 식으로 표현하면 식(16)과 같으며 이에 대한 진리치표는 표2와 같다.

$$b_j = \begin{cases} Y_{0j} & \text{if } T_1T_0=00 \ : \text{가산기 모듈} \\ Y_{1j} & \text{if } T_1T_0=01 \ : \text{감산기 모듈(생략가능)} \\ Y_{2j} & \text{if } T_1T_0=10 \ : \alpha^r\text{생성 모듈} \\ Y_{3j} & \text{if } T_1T_0=10 \ : \text{승법역원생성 모듈} \end{cases} \quad (16)$$

표 2. 배분기 모듈 D_2의 기본 셀의 진리치표
Table 2. Truth table for basic D_2-cell of distribution module D2.

b_j	T_1	T_0	G_{D20}	G_{D21}	Y_{0j}	G_{D22}	G_{D23}	Y_{2j}
b_j	0	0	ON	ON	b_j	ON	OFF	—
b_j	0	1	ON	OFF	—	ON	ON	b_j
b_j	1	0	OFF	ON	—	OFF	OFF	—
b_j	1	1	OFF	OFF	—	OFF	ON	—

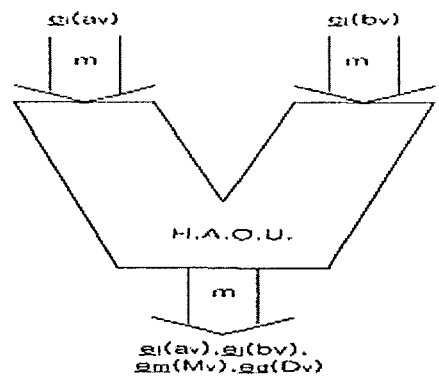
continued

b_j	T_1 T_0	G_{D24} G_{D25} Y_{0j}	G_{D26} G_{D27} Y_{2j}
b_j	0 0	OFF ON −	OFF OFF −
b_j	0 1	OFF OFF −	OFF ON −
b_j	1 0	ON ON −	ON OFF −
b_j	1 1	ON OFF −	ON ON −

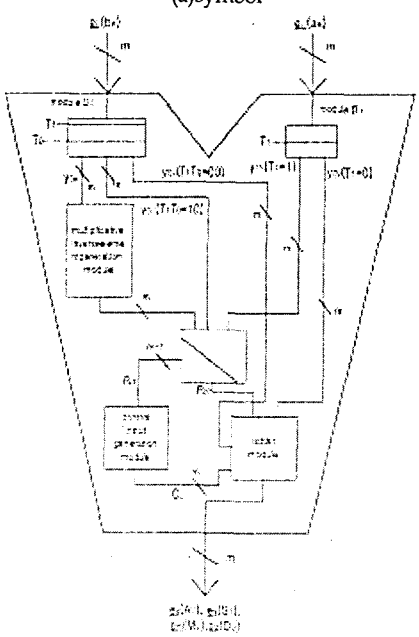
where, − means nonpass.

4-3. 고속 연산기 구성

앞의 1절과 2절의 내용을 토대로 최종 고속 연산기를 구성하면 다음 그림1과 같다.



(a)symbol



(b) internal structure

그림 1. 고속 연산기시스템 블록도

Fig. 1. Block diagram of highly efficiency arithmetic operation unit system

V. 결 론

본 논문에서는 최근에 그 활용과 향 후 지식기반의 정보화시대에 그 활용도가 증가되고 있는 각종 멀티미디어 하드웨어시스템에 반드시 필요한 고속 연산기 구성의 한가지 방법을 제안하였다.

제안한 고속 연산기는 Galois체 $GF(2^m)$상에서 구성하였다.

특히, 가산기 모듈은 가산, 감산, 승산, 제산의 어떤 연산을 하더라도 항상 사용된다. 그리고 가산기 모듈내의 cell-A의 개수는 m개, α^r생성 모듈내의 cell-M의 개수는 m^2개이고 분배기 모듈 D_1과 D_2내의 D_1-cell과 D_2-cell의 개수는 각각 m개이다.

또한, 제안한 고속 연산기는 모듈들의 합성으로 구성되므로 m의 확장에 따른 고속 연산기는 각 모듈을 m에 따라 확장만 하면 되며 최종 고속 연산기는 분배기 모듈로서 합성하여 용이하게 구성할 수 있다.

이상의 내용을 종합하면 현재. 사용하고 있는 디지털시스템 및 스위칭이론을 그대로 적용할 수 있는 장점이 있으며 기존의 컴퓨터를 $GF(2^m)$상에서 m의 확장에 따라 용이하게 확장할 수 있다.

참고문헌

[1] S.Y.Kung, *VLSI ARRAY PROCESSORS*, Prentice-hall,Inc.,1988.

[2] M.D.Ercegovac and T.Lang, *Digital Systems and Hardware/Firmware Algorithms*, John Wiley & Sons, Inc., Canadea, 1985.

[3] L.Rojas-Cardenas, P.Senac,L.Dairaine and M.Diaz,"Temporal Partial Order and Partial Reliablity setvice for Distributed Multimedia Applacations", MMM'98.

[4] I.F.Blake,*Algebraic Coding Theory:History and Development*, Down, Hutchinson & Ross,Inc.,Stroudburg,Pennsylvania,1973.

[5] R.Lidi and G.Pilz,*Applied Abstract Algebra* ,Spring-Verlg,Inc.,N.Y.,1984.

[6] C.S.Yeh, I.S.Reed and T.K.Trung,"Systolic multiplier for finite filds $GF(2^m)$,"IEEE Trans.Comput.,vol.C-33,pp.357-360,Apr.1984.

[7] K.ZPekmastzi,"Multiplexer-Based Array Multipliers", IEEE Trans Comput., vol.48 NO.1,pp.15-23.Jan.1999.

[8] G.Drolet,"A New Representation of Elements of Finite Fields $GF(2^m)$ Yields Small Complexity Arithmetic Circuits," IEEE Trans Comput., Vol.47 NO.9,pp.938-946, Sep.1998.

[9] E.Artin,*Galois Theory*,NAPCO Graphic arts,Inc.,Wisconsin.1971.