論文 98-23-5-2

유한체 $GF(2^m)$ 상의 역원계산 회로 및 나눗셈 회로 설계

正會員 조 용 석*, 박 상 규**

Design of Inversion and Division Circuit over $GF(2^m)$

Yong Suk Cho*, Sang Kyu Park** *Regular Members*

요 약

본 논문에서는 유한체 $GF(2^m)$ 상의 새로운 역원계산 알고리듬을 제안하고 이를 이용한 역원계산 회로 및 나눗셈 회로를 설계한다. 제안된 역원계산 알고리듬은 Fermat의 정리에 기초한 것으로 약 $m/2$개의 clock cycle에 역원을 구할 수 있다. 본 알고리듬을 이용하여 설계한 $GF(2^m)$ 상의 역원계산 회로 및 나눗셈 회로는 멀티플렉서 이외에 다른 부가 하드웨어가 필요하지 않으므로 매우 간단한 하드웨어로 구현할 수 있는 장점을 가지고 있다.

ABSTRACT

In this paper, we propose a new algorithm for computing multiplicative inverses in $GF(2^m)$ and design an inversion circuit and a division circuit using this algorithm. The algorithm used is based on Fermat's theorem. It takes around $m/2$ clock cycles. The hardware requirements of the inversion circuit and the division circuit using this algorithm are the same as traditional circuits except for the addition of multiplexers.

I. 서 론

유한체(finite fields or Galois fields) 상의 연산은 오류정정 부호, 디지털 신호처리, 암호화 등의 여러 분야에서 널리 사용되고 있다. 특히 오류정정 부호 중 실용 상 널리 사용되고 있는 BCH 부호와 Reed-Solomon 부호와 같은 블럭부호는 유한체 상에서 정의되며 모든 연산이 유한체 상에서 이루어진다. 따라서 유한체 상의 연산은 이들 부호의 부호기 및 복호기 설계 시 전체 시스템의 규모나 성능에 절대적인 영향을 미친다[1].

유한체 $GF(2^m)$에서 정의되는 덧셈(addition), 뺄셈(subtraction), 곱셈(multiplication), 나눗셈(division)의 4칙 연산 중에서 가장 어렵고 복잡한 연산이 나눗셈이다. 일반적으로 유한체 $GF(2^m)$ 상의 임의의 두 원소 사이의 나눗셈은 한 원소의 곱셈 역원(multiplicative inverse, 이하 역원으로 표기)에 다른 원소를 곱하는 두 단계로 수행된다.

유한체 $GF(2^m)$에서 0이 아닌 임의의 한 원소의 역원을 계산하는 가장 간단한 방법은 ROM(Read Only Memory)을 이용한 표참조(table lookup) 방법이다[2].

*영동대학교 전자공학부
**한양대학교 공과대학 전자전기공학부
論文番號:98025-0115
接受日字:1998年 1月 15日

그러나 이 방법은 유한체의 크기가 작을 때는 효율적이지만 유한체의 크기가 커지면 ROM의 용량($2^m \times m$ 비트)이 급격하게 증가하는 문제점이 있다. 알고리듬적으로 역원을 계산하는 방법에는 Fermat의 정리를 이용한 방법[3]과 Euclid 알고리듬을 이용한 방법[4] 등이 있다.

본 논문에서는 Fermat의 정리를 이용한 새로운 역원계산 알고리듬을 제안한다. 일반적으로 Fermat의 정리를 이용한 역원 계산 알고리듬은 $m-2$개의 clock cycle이 필요하고 이 동안에 $m-2$번의 곱셈을 수행하게 된다[3]. 최근 Fenn[5] 등은 $m/2$개의 clock cycle만에 역원을 계산할 수 있는 고속 역원계산 알고리듬을 제안하였다. 그러나 이 방법은 $x \times x^k$를 생성하는 부가적인 하드웨어를 필요로 한다. 이 $x \times x^k$를 생성하는 하드웨어는 유한체 $GF(2^m)$ 상의 병렬 곱셈기와 거의 같은 하드웨어 복잡도를 가진다. Calvo와 Torres[6]는 이러한 부가적인 하드웨어가 필요 없는 알고리듬을 제안하였다. 그러나 이 방법에서는 부가적인 하드웨어가 필요 없는 대신에 복잡한 제어신호가 필요하고 그 만큼 많은 클럭 시간이 필요하게 되는 단점을 가지고 있다.

본 논문에서는 $GF(2^m)$에서 m이 짝수인 경우 $m/2$번, m이 홀수인 경우 $(m+1)/2$번 의 곱셈을 수행하여 역원을 계산할 수 있는 새로운 방법을 제안하고 이를 이용한 역원계산 회로를 설계한다. 제안된 방법에 의해 설계된 역원계산 회로는 $2m$개의 멀티플렉서(multiplexer)만을 부가하고 이를 제어하는 하나의 제어신호만을 사용하므로 문헌 [5]와 [6]의 방법에 비해 매우 간단하게 구현할 수 있는 장점이 있다. 또한 설계된 역원계산 회로에 m개의 멀티플렉서를 추가하고 한 클럭을 더 사용하여 나눗셈을 수행할 수 있는 나눗셈 회로를 설계한다.

먼저 Ⅱ에서는 기존의 Fermat의 정리에 기초한 역원계산 알고리듬을 살펴보고 새로운 고속 역원계산 알고리듬을 제안한다. Ⅲ에서는 제안한 알고리듬을 기초로 기존의 회로에 비해 매우 간단한 역원계산 회로 및 나눗셈 회로를 설계한다. 끝으로 Ⅳ에서 결론을 맺는다.

Ⅱ. 유한체 $GF(2^m)$ 상의 역원계산 알고리듬

유한체 $GF(2^m)$은 2^m개의 원소를 가지고 있으며 그 원소들은 2진 계수를 갖는 $m-1$차 이하의 다항식으로 표현할 수 있다. 즉 유한체 $GF(2^m)$은 2진체(binary field)인 $GF(2)$ 상의 m차원 벡터공간(vector space)이 된다. 여기에서 선형독립인 m개의 벡터는 모두 기저(basis)가 된다. 여러 가지 유용한 기저 중에서 정규기저(normal basis)[3]는 제곱(squaring) 연산이 한 비트 순회 쉬프트(cyclic shift)만으로 수행되는 장점을 가지고 있다. 따라서 본 논문에서는 이 정규기저 상에서의 연산을 가정한다.

유한체 $GF(2^m)$ 상의 0이 아닌 임의의 한 원소를 x라 하면 Fermat의 정리에 의해

$$x^{2^m} = x \tag{1}$$

가 된다. 따라서 x의 역원 x^{-1}은 다음과 같이 쓸 수 있다.

$$x^{-1} = x^{2^m-2} \tag{2}$$

여기에서 $2^m-2 = 2+2^2+2^3+\cdots+2^{m-1}$이므로 식 (2)를 다시 쓰면

$$x^{-1} = x^{2^m-2} = x^2 \cdot x^{2^2} \cdot x^{2^3} \cdots x^{2^{m-1}} \tag{3}$$

가 된다. 따라서 식 (3)을 이용하면 연속적인 순회 쉬프트와 $m-2$번의 곱셈으로 역원을 계산할 수 있다. 예를 들어 $m=8$인 경우 식 (3)은 다음과 같이 된다.

$$x^{-1} = x^{254} = x^2 \cdot x^4 \cdot x^8 \cdot x^{16} \cdot x^{32} \cdot x^{64} \cdot x^{128} \tag{4}$$

식 (3)을 이용하면 그림 1과 같은 역원계산 회로를 설계할 수 있다. 그림 1에서 ⊠는 $GF(2^m)$ 상의 병렬 곱셈기이며 $()^2$는 $GF(2^m)$ 상의 제곱기이다. 정규기저 상에서의 제곱기는 한 비트 순회 쉬프트 시키면 되므로 별도의 하드웨어가 필요 없이 결선만 바꾸어 주면 된다.

그림 1과 같은 역원계산 회로는 초기에 레지스터 A에 x^2을 레지스터 B에 x^4을 로드하면 $m-2$ 클럭 후에 x의 역원인 x^{-1}의 값이 출력에 나타나게 된다. 예를 들어 $m=8$인 경우 그림 1의 회로의 동작을 표 1에

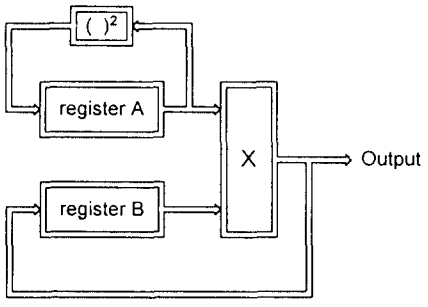


그림 1. $GF(2^m)$ 상의 역원계산 회로
Fig. 1 Inversion circuit for $GF(2^m)$

표 1. $m=8$인 경우 그림 1 회로의 동작상태
Table 1. Values held in registers in Fig. 1 for $m=8$

클럭	레지스터 A	레지스터 B	출 력	제어동작
0	x^4	x^2	x^2x^4	병렬 로드
1	x^8	x^2x^4	$x^2x^4x^8$	
2	x^{16}	$x^2x^4x^8$	$x^2x^4x^8x^{16}$	
3	x^{32}	$x^2x^4x^8x^{16}$	$x^2x^4x^8x^{16}x^{32}$	
4	x^{64}	$x^2x^4x^8x^{16}x^{32}$	$x^2x^4x^8x^{16}x^{32}x^{64}$	
5	x^{128}	$x^2x^4x^8x^{16}x^{32}x^{64}$	$x^2x^4x^8x^{16}x^{32}x^{64}x^{128}=x^{-1}$	

나타내었다.

식 (3)은 다음과 같이 다시 쓸 수 있다.

$$x^{-1}=\begin{cases} x\times\prod_{i=0}^{(m-3)/2}[x^{12}]^{2^{2i}} & m \text{ odd} \\ x^2\times\prod_{i=0}^{(m-4)/2}[x^{12}]^{2^{2i}} & m \text{ even} \end{cases} \quad (5)$$

식 (5)를 이용하면 $x^{12}(=x^4\times x^8)$를 구하는데 필요한 1번의 곱셈을 포함하여 m이 짝수인 경우 $m/2$번, m이 홀수인 경우 $(m+1)/2$번의 곱셈으로 역원을 계산할 수 있다.

예를 들어 $m=8$인 경우 식 (5)는 다음과 같이 된다.

$$x^{-1}=x^2\cdot(x^{12})^{2^0}\cdot(x^{12})^{2^2}\cdot(x^{12})^{2^4}=x^{2^8-2} \quad (6)$$

따라서 $8/2=4$번의 곱셈으로 역원을 구할 수 있다. 또한 $m=9$인 경우 식 (5)는

$$x^{-1}=x\cdot(x^{12})^{2^0}\cdot(x^{12})^{2^2}\cdot(x^{12})^{2^4}\cdot(x^{12})^{2^6}=x^{2^9-2} \quad (7)$$

가 되므로 $(9+1)/2=5$번의 곱셈으로 역원을 구할 수 있다.

따라서 식 (5)를 이용하여 역원을 계산하면 식 (3)을 사용하였을 경우보다 더 고속으로 역원을 계산할 수 있게 된다.

Ⅲ. 고속 역원계산 회로 및 나눗셈 회로 설계

식 (5)를 살펴보면 항상 x^{12}을 4승씩 하면서 곱해나감을 알 수 있다. x^{12}는 $(x\times x^2)^4$으로 구할 수 있다. 이러한 성질을 이용하면 그림 2와 같은 고속 역원계산 회로를 설계할 수 있다.

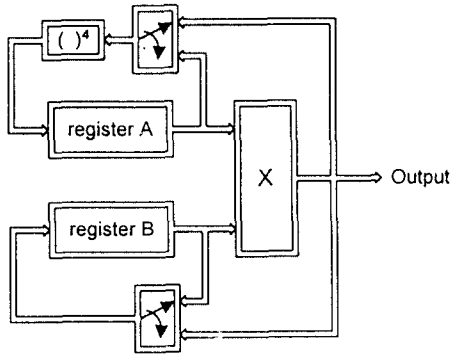


그림 2. $GF(2^m)$ 상의 고속 역원계산 회로
Fig. 2 Fast inversion circuit for $GF(2^m)$

그림 2에서 $(\ )^4$는 $GF(2^m)$ 상의 4승 회로로, 정규기저 상에서는 2비트 순회 쉬프트 시키면 되므로 별도의 하드웨어가 사용되지 않는다. 스위치는 각각 m개의 2×1 멀티플렉서로 구현할 수 있다.

그림 2와 같은 역원계산 회로의 동작을 살펴보면 다음과 같다. 먼저 m이 짝수인 경우 초기에 레지스터 A에는 x를 레지스터 B에는 x^2을 로드시킨다. 그 다음 클럭이 인가되면 레지스터 A에는 $(x\times x^2)^4=x^{12}$가 입력되고 레지스터 B에는 다시 x^2이 입력된다. 이 순간에 스위치를 아래 방향으로 절체한다. 그러면 그 다

음 클럭부터는 레지스터 A에는 $(x^{12})^4$가 입력되고 레지스터 B에는 병렬 곱셈기의 출력이 입력된다. 따라서 $m/2$ 클럭 후에 x의 역원인 x^{-1}이 출력된다. 예를 들어 $m=8$인 경우 그림 2의 회로의 동작상태를 표 2에 나타내었다.

m이 홀수일 경우에는 초기에 레지스터 A에 x^2을 레지스터 B에 x를 로드시키는 것 외에는 짝수인 경우와 동일하게 동작한다. m이 홀수인 경우는 $(m+1)/2$ 클럭 후에 x의 역원인 x^{-1}이 출력된다. 예를 들어 $m=9$인 경우 그림 2의 회로의 동작상태를 표 3에 나타내었다.

표 2. $m=8$인 경우 그림 2 회로의 동작상태
Table 2. Values held in registers in Fig. 2 for $m=8$

클럭	레지스터 A	레지스터 B	출 력	제어동작
0	x	x^2	x^3	병렬 로드
1	$(x^3)^4=x^{12}$	x^2	$x^2\,(x^{12})$	스위치 절체
2	$(x^{12})^4$	$x^2\,(x^{12})$	$x^2\,(x^{12})(x^{12})^4$	
3	$(x^{12})^{16}$	$x^2\,(x^{12})(x^{12})^4$	$x^2\,(x^{12})(x^{12})^4(x^{12})^{16}=x^{-1}$	

표 3. $m=9$인 경우 그림 2 회로의 동작상태
Table 3. Values held in registers in Fig. 2 for $m=9$

클럭	레지스터 A	레지스터 B	출 력	제어동작
0	x^2	x	x^3	병렬 로드
1	$(x^3)^4=x^{12}$	x^2	$x^2\,(x^{12})$	스위치 절체
2	$(x^{12})^4$	$x^2\,(x^{12})$	$x^2\,(x^{12})(x^{12})^4$	
3	$(x^{12})^{16}$	$x^2\,(x^{12})(x^{12})^4(x^{12})^{16}$		
4	$(x^{12})^{64}$	$x^2\,(x^{12})(x^{12})^4(x^{12})^{16}$	$x^2\,(x^{12})(x^{12})^4(x^{12})^{16}(x^{12})^{64}=x^{-1}$	

Calvo와 Torres가 제안한 방법에서는 2개의 레지스터 A, B와 병렬곱셈기 한 개를 사용하여 먼저 레지스터 A에 x를 로드하고 한 비트 순회 쉬프트(cyclic shift)한 x^2을 레지스터 B로 전송한다. 그 다음 레지스터 A를 다시 한 비트 순회 쉬프트하여 x^4을 생성한 다음 레지스터 B와 곱하여 x^6을 구하고 이를 다시 레지스터 A로 복사한다. 그 다음부터는 레지스터 A를 두 비트씩 쉬프트하여, $m=8$인 경우 다음과 같이 역원을 구한다.

$$x^{-1}=x^{2^7}\cdot(x^6)^{2^6}\cdot(x^6)^{2^2}\cdot(x^6)^{2^4}=x^{2^8-2} \tag{8}$$

따라서 x^6을 구하는데 여러 번의 클럭이 필요하게 되고 또 레지스터간의 데이터 전송을 위한 제어신호가 필요하게 되는 단점이 있다.

그러나 본 논문에서 제안한 방법은 그림 2와 같이 $2m$개의 멀티플렉서와 이들을 제어하는 하나의 제어신호만으로 고속의 역원계산 회로를 설계할 수 있다.

나눗셈은 한 원소의 역원에 다른 원소를 곱하면 되므로 나눗셈 회로는 그림 2와 같은 회로에 $GF(2^m)$의 병렬 곱셈기를 추가하면 쉽게 구현할 수 있다. 그러나 $GF(2^m)$의 병렬 곱셈기는 많은 하드웨어가 소요된다. 따라서 그림 3과 같이 m개의 멀티플렉서와 한 클럭을 더 사용하여 나눗셈을 수행하는 것이 하드웨어를 절약할 수 있다.

예를 들어 그림 3과 같은 나눗셈 회로에서 $z=y/x$를 계산한다면 $z=yx^{-1}$이므로 레지스터 C에 y를 기억시켜 놓고 먼저 m이 짝수인 경우에는 $m/2$ 클럭 동안 또는 m이 홀수인 경우에는 $(m+1)/2$ 클럭 동안에 x의 역원 x^{-1}을 계산한 다음 스위치 SW3을 아래 방향으로 절체하면 그 다음 클럭에 나눗셈의 결과인 z를 얻을 수 있다.

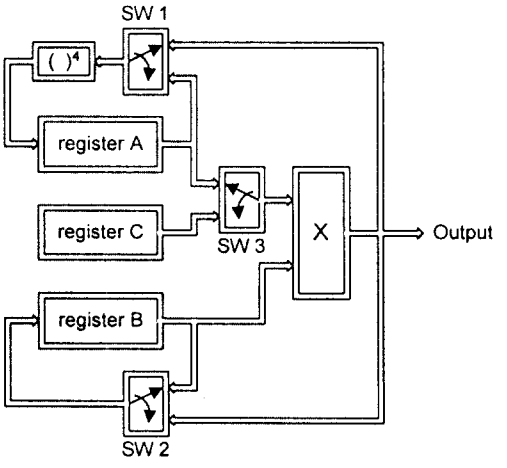


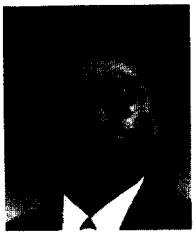
그림 3. $GF(2^m)$ 상의 고속 나눗셈 회로
Fig. 3 Fast division circuit for $GF(2^m)$

Ⅳ. 결 론

본 논문에서는 Fermat의 정리를 이용한 고속의 역원계산 알고리듬을 제안하고 이를 이용한 역원계산 회로와 나눗셈 회로를 설계하였다. 제안된 알고리듬은 m이 짝수인 경우 $m/2$번, m이 홀수인 경우는 $(m+1)/2$번의 곱셈으로 역원을 계산할 수 있다. 제안된 방법을 이용하여 설계한 역원계산 회로는 $2m$개의 멀티플렉서(multiplexer)만을 부가하고 이를 제어하는 하나의 제어신호만을 사용하므로 문헌 [5]와 [6]의 방법에 비해 매우 간단하게 구현할 수 있는 장점을 가지고 있다. 또한 설계된 역원계산 회로에 m개의 멀티플렉서를 추가하고 한 클럭을 더 사용하여 나눗셈을 수행할 수 있는 나눗셈 회로를 설계하였다.

참 고 문 헌

1. M. Y. Rhee, *Error Correcting Coding Theory*, McGraw-Hill, New York, 1989.
2. S. B. Wicker and V. K. Bhargava, *Reed-Solomon Codes and Their Applications*, IEEE Press, 1994.
3. C. C. Wang, T. K. Truong, H. M. Shao, L. J. Deutsch, J. K. Omura, and I. S. Reed, "VLSI Architectures for Computing Multiplications and Inverses in $GF(2^m)$," *IEEE Trans. Computers*, vol. 34, no. 8, pp. 709~716, Aug. 1985.
4. K. Araki, I. Fujita, and M. Morisue, "Fast Inverter over Finite Field Based on Euclid's Algorithm," *Trans. IEICE E-72*, pp. 1230~1234, Nov. 1989.
5. S. T. J. Fenn, M. Benaissa, and D. Taylor, "Fast Normal Basis Inversion in $GF(2^m)$," *Electronics Letters*, vol. 32, pp. 1566~1567, Aug. 1996.
6. I. J. Calvo and M. Torres, "Complexity of the Inversion in $GF(2^m)$," *Electronics Letters*, vol. 33, pp. 194~195, Jan. 1997.



조 용 석(Yong Suk Cho) 정회원

1986년 2월 : 한양대학교 공과대학 전자통신과(공학사)
1988년 2월 : 한양대학교 대학원 전자통신과(공학석사)
1997년 2월 : 한양대학교 대학원 전자통신과 박사과정 수료
1989년 4월~1996년 2월 : 한국통신 연구개발원
1996년 3월~현재 : 영동대학교 전자공학부 조교수
※주관심분야 : 부호이론, 디지털통신, 확산대역통신, PCS, IMT-2000
e-mail : yscho@kachi.yit.ac.kr

박 상 규(Sang Kyu Park) 정회원

1974년 2월 : 서울대학교 전기공학(공학사)
1980년 5월 : Duke University 통신공학(공학석사)
1987년 5월 : University of Michigan 통신공학(공학박사)
1976년 7월~1978년 10월 : 국방과학연구소
1990년 8월~1991년 8월 : University of Southern California 객원교수
1987년 3월~현재 : 한양대학교 공과대학 전자전기공학부 교수
※주관심분야 : 디지털통신, 확산대역통신, 부호이론, PCS, IMT-2000
e-mail : skpark@email.hanyang.ac.kr