정보학연구 제5권 제4호 2002년 12월(pp.145-154)
The Journal of Information Technology, Vol.5, No.4

CRT를 사용한 잉여수계 기수확장에 관한 연구

김용성**

요 약

기수확장(Extension of Base)은 잉여수계(RNS:Residue Number System)에서 모듈리를 확장하기 위한 기본적인 방법이다. 잉여수계는 병렬성과 모듈간의 자리올림 수의 전달이 없는 장점을 갖지만, 기수확장 등에 의해 전체적인 시스템의 성능이 저하되며, 혼합기수 변환을 적용한 기존의 방법에서는 연산기의 크기는 감소하지만 연산속도가 저하되는 문제점을 갖는다.
그러므로, 본 논문에서는 CRT를 사용한 개선된 기수확장을 수행하여, 비교적 적은 크기이며, 속도가 향상된 기수 확장기를 설계할 수 있었다.

Ⅰ. 서론

기수확장(Extension of Base)은 잉여수계(RNS: Residue Number System)의 연산 시 연산 적용 범위의 확장이 요구되는 경우에 주로 사용하며, 일반적으로 주 연산 후에 기수를 확장을 하기 위하여 사용한다.[1] 잉여수계 연산은 모듈리(moduli) 간의 자리올림수의 전달이 필요 없고, 병렬구조로 수행되므로, 디지털 신호처리 및 그래픽, 신경망 등과 같은 전용 프로세서 설계 시에 효율적인 설계 방법이다.[2] [6] 그러나 일반적인 제산과 같은 경우에 문제점을 가지므로, 스케일링(Scailing)에 의해서 고정된 수에 대한 제산을 주로 사용하며, 스케일링 후 기수확장을 수행한다. 그러므로 기수확장은 프로세서의 전체 실행시간을 감소시키는 요인으로 작용된다. Szabo, Richard I. Tanaka에 의해서 제안된 기본적인 방법과 이를 고속화한 방법에서도 혼합기수변환기(MRC: Mixed Radix Converter)을 주로 사용하므로[1][7-8], p개의 잉여수에 대한 스케일링 후, 기수확장을 수행하는 경우 병렬적으로 처리되지만 N-p-1(N: 모듈리 수)단계를 수행하게 되며, 스케일링에 적용한 제수의 개수에 반비례하여 기수 확장의 연산 단계가 변화하므로 수행 속도가 달라지게 되는 문제점을 갖는다. Barsi에 의한 변형된 CRT(Chinese Remainder Theorem)를 사용한 기수확장은 일정한 수행속도로 기수확장을 수행할 수 있지만[9], LUT (Look Up Table)연산기의 크기가 증가하며, 모듈리 수가 작은 경우에도 동일한 방법을 적용함으로써 연산속도가 저하되는 문제점을 갖는다.

그러므로, 본 논문에서는 모듈리의 수 및 모듈러스(Modulus)의 크기에 따라 CRT를 사용한 개선된 기수 확장기를 설계하여 연산기의 크기 및 연산속도를 향상시키고자 한다.

* 여주대학 컴퓨터사이언스과 부교수

II. 잉여수계와 MRC를 사용한 기수확장

잉여수계 연산에서는 사용할 정수의 최대 값에 따라 모듈리를 P선택하고, 각각의 모듈러스는 식(1)과 같이 표현되며, 정수 Y의 최대 값은 식(2)와 같이 각 모듈러스의 곱으로 표현된다.[1]

$$P = \{ m_1, m_2, \ldots, m_n \} \quad (1)$$

$$0 \le Y_{\max} < M, \quad M = \prod_{i=1}^{n} m_i \quad (2)$$

이 때, 정수 Y의 잉여수 표현은 다음과 같이 n개 터플(tuple)로 구성되고,

$$Y \xrightarrow{RNS} (|y|_{m_1}, |y|_{m_2}, \ldots, |y|_{m_n})$$

(단, $|y|_{m_i} = y \bmod m_i$)

이항연산(Binary operation)은 식 (3)과 같이 표현된다.

$$|d|_{m_i} = |x|_{m_i} \circ |y|_{m_i} \quad (i=1,\ldots,n : n:\text{정수}) \quad (3)$$

잉여수계의 제산은 일반적인 잉여수계 제산이 갖는 속도문제 및 복잡도로 인하여 곱의 역 및 스케일링(Scaling)을 주로 사용하게 된다. 나머지 없는 제산은 식(4)와 같이 곱의 역(Multiplicative Inverse)를 사용하며,

$$|1//y|_{m_i} = |d|_{m_i},$$

(단, $| |d|_{m_i} \cdot |y|_{m_i} |_{m_i} = 1$) (4)

젯수가 x이고 피젯수가 y일 때 몫 Q는 다음 식과 같이 표시된다.

$$|Q|_{m_i} = | |x|_{m_i} \cdot |1//y|_{m_i} |_{m_i} = | x//y |_{m_i}$$

(단, x,y는 정수, y와 m_i는 서로소,'//': 곱의 역)

스케일링은 주어진 모듈리를 사용하여 연산한다. 정수 x를 모듈리 m_i로 스케일링하는 경우,

$x = \left[\frac{x}{m_i} \right] m_i + |X|_{m_i}$(단, "[ ]"는 몫의 정수부)

로 표현되므로, $m_1, \ldots, m_k$까지 k개의 모듈리로 스케일링하는 경우에 대한 스케일링 결과는 식(5)와 같이 표현된다.

$$s(0) = \left[\frac{x}{m_1} \right] = \frac{x - |x|_{M_1}}{m_1}$$

$$s(k) = \left[\frac{s(k-1)}{m_k} \right] = \frac{s(k-1) - |s(k-1)|_{M_k}}{m_k},$$

(k 는 정수, k<n) (5)

식(5)에서 m_1으로 스케일링할 때, 모듈러스 m_1에서 $|X - \langle X \rangle m_1|_{m_1} = 0$이고 $|1// m_1|_{m_1}$은 성립되지 않으므로, 모듈러스 m_1에서 스케일링된 잉여수 값은 구할 수가 없으며, 다른 모듈리의 결과를 사용하여 기수확장(Extension of Base)을 수행하여 산출한다. 스케일링된 결과 S를 모듈리 $P=\{ m_1, m_2, \ldots, m_n \}$를 사용하여 표시하는 경우, 새로운 모듈러스 m_B에 대한 잉여수를 구하는 기수확장의 기본적인 방법은 식(6-1)과 같은 혼합기수 변환(MRC: Mixed Radix Conversion)을 사용한다.[1][8] 스케일링된 결과에 대한 혼합기수 변환을 식(6-1)로 표현하였을 때, 모듈러스 m_B에 대한 기수 확장은 식(6-2)로 표현하면, $a_B = 0$가 되므로, 혼합기수

변환과 동일한 방법을 적용하여 m_B에 대한 잉여수를 산출한다.(식(6-2)과 식(6-1)의 m_i는 동일 순서이어야 함.)

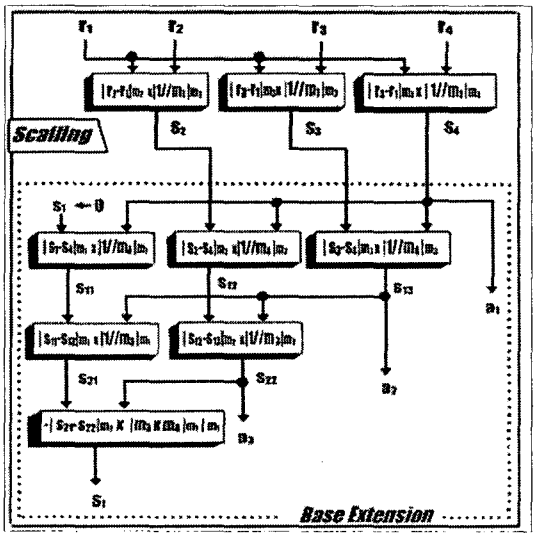
$$s = an\prod_{i=1}^{N-1} m_i + \ldots + a_2 m_1 + a_1 \quad (6\text{-}1)$$

$$s = a_B \times m_B \times \prod_{i=1}^{N} m_i + a_n \prod_{i=1}^{N-1} m_i + \ldots + a_2 m_1 + a_1$$

, $a_k=0$ (6-2)

모듈 리가 $m_1, \ldots, m_4$ 이고, $r_1,\ldots, r_4$인 잉여수를 m_1로 스케일링을 수행하는 경우, 기수확장이 포함된 LUT(Look Up Table) 스케일링 연산기를 그림 1에 나타내었다.

그림 1에서 첫 번째 단은 식(5)의 기본적인 스케일링 연산을 수행하며, 첫 번째 단의 모듈



(그림 1) 기수확장이 포함된 LUT 스케일링 연산(모듈리 4개)

(Fig 1) LUT Scaling operation with Extension of Base(No. of Moduli=4)

러스 m_i에서 스케일링된 결과를 s_i라 할 때, 두 번째 단 이 후는 식 (6-2)에 따라 식(7)의 과정을 수행하여 기수 확장된 모듈리 S_1을 산출한다. (단, 모듈러스는 $m_4,\ldots, m_1$순서로 적용함)

$$s = (a_5 m_1 m_2 m_3 + a_4 m_2 m_3 + a_3 m_3 + a_2) m_4 + a_1,$$
$$a_1 = |s|_{m_4} = s_4$$
$$s_{1j} = |(s - a_1)/m_4|_{m_j} \ (j=1,..,3), \quad a_2 = |s_{13}|_{m_3}$$
$$s_{2j} = |(s_{1j} - a_2)/m_3|_{m_j} \ (j=1,2), \quad a_3 = |s_{22}|_{m_2}$$
$$s_{31} = |(s_{2j} - a_3)/m_2|_{m_1}, \quad a_4 = |s_{31}|_{m_1} \quad (7)$$

즉, 그림 1의 s_1의 초기치를 0으로 하고, $a_4=0$임으로, 식(7)에 따라 연산하면, 마지막 단에서 $|s_1 + (s_{2j} - a_2))/(m_4 \times m_3)|_{m_1} = 0$에 의하여 다음 식과 같이 기수 확장된 출력을 생성한다.

$$|s_1|_{m_1} = |-(s_{2j} - a_2)(m_4 \times m_3)|_{m_1}$$

혼합기수 연산을 사용한 기수확장은 모듈리수 N개에 대하여 1개의 기수확장을 수행하는 경우, N-1단계의 시간이 소요된다.

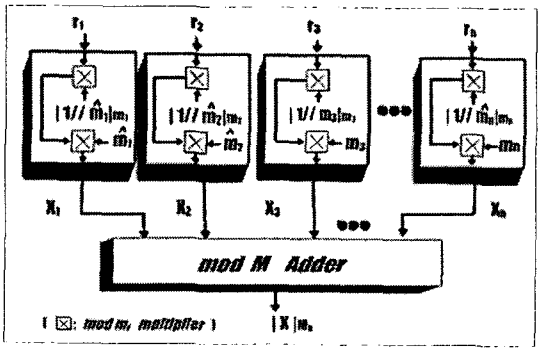
III. CRT를 사용한 기수확장

일반적으로 혼합기수 변환은 II절에서와 같이 스케일링, 오버플로우 검출, 부호 검출 등에 주로 사용하고, CRT는 비가중치 수 체계(Unweighted Number System)를 가중치 수 체계(Weighted Number System)로 전환하기 위하여 사용하며, 식(8)과 같이 표시된다.[9]

$$X_i = \bigwedge_{m_i} |r_i // \bigwedge_{m_i} |m_i , \quad X_N = \sum_{i=1}^{N} X_i ,$$
$$X = |X_N|_M \qquad (8)$$
(단, x: 정수, $0 \le X \le M$,
$$M = \prod_{i=1}^{N} m_i, \quad \bigwedge_{m_i} = M//m_i)$$

식(8)에서 전환된 결과 $|X|_M$에 모듈리 m_B에 대한 잉여수를 구하면 모듈리 m_B에 대한 잉여수를 구할 수 있다. 그러나 CRT를 연산과정은 그림 2와 같이 승산기 및 가산기로 연산이 이루어지고 , 식(8)에서 X_s의 값의 범위가 식(9)와 같이되므로 자리올림수 전달문제 등으로 연산속도가 저하되며, 잉여수계 연산의 장점을 상실되는 문제점을 갖게 된다.

$$0 \le X_s < NM - \sum_{i=1}^{N} \bigwedge_{m_i} \qquad (9)$$



(그림 2) 산술 연산기를 사용한 CRT 생성기
(Fig. 2) CRT generator using Arithmatic operator

이러한 문제점을 해결하기 위하여 Barsi가 제안한 CRT 연산은 식(10-2)의 $M^{(p)}$값을 사용하여 식(10-1)과 같이 $a_{i,p}$정의하여 CRT 연산을 식(11)과 같이 나타내면, X_E의 범위는 $0 \le X_E < 2M$ 가 되어, CRT 연산 시 사용되는 수의 범위를 줄일 수가 있다.

$$a_{i,p} = [(\bigwedge_{m} |x_i / \bigwedge_{m} |_{m_i}) / M^{(p)}] \qquad (10\text{-}1)$$

$$M^{(p)} = \prod_{j=1}^{p} m_j \quad (1 \le p < n),$$
$$(0 \le a_{i,p} \le (M/M^{(p)}) - (\bigwedge_{m} / M^{(p)}) \qquad (10\text{-}2)$$

$$X_E = M^{(p)} | \sum_{i=1}^{n} a_{i,p} |_{M/M(p)} + \sum_{i=1}^{p} M^{(P)}/m_i \; |M/M^{(p)} \; |x_i/M_i|_{m_i}|_{m_i} \qquad (11)$$

X_E가 M보다 큰 경우에 대해서는 식(12)의 조건 식에 따라 감산을 수행하여 최종적인 X값을 산출하고, 이 결과에 모듈리 M_B에 대한 잉여수를 구하여 식 (13)과 같이 기수확장을 수행한다.

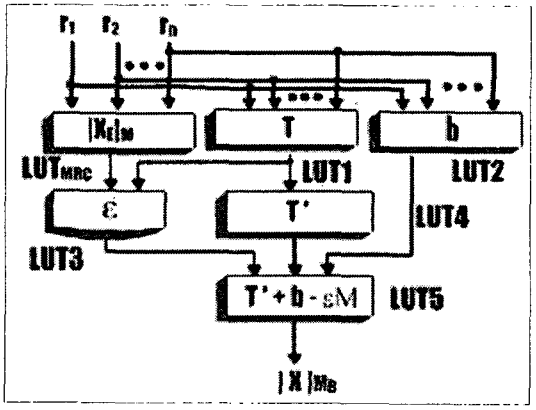
조건 1:
$$| \sum_{i=1}^{n} a_{i,p} |_{M/M(p)} \ge (M/M^{(p)}) - p + 1$$
조건2 :
$$|X_E|_M \le (p-1)M^{(p)} - M^{(p)} \sum_{i=1}^{p} (1/m_i)$$
, if 1),2) then $\varepsilon = 1$ (12)

$$X_B = | M^{(p)} | \sum_{i=1}^{n} a_{i,p} |_{M/M(p)} + \sum_{i=1}^{p} (M^{(P)}/m_i) \; |M/M^{(p)} |x_i/M_i|_{m_i}|_{m_i} - \varepsilon M |_{m_n} \qquad (13)$$

식(13)에서 연산기 구성에 따라 아래와 같이 T, T', b, b_i로 분리하여 표기하고, 이에 대한 기수확장을 그림 3에 나타내었으며, 각 연산기는 LUT를 사용하였다. 그림3에서 T 및 b를 연산하기 위하여 LUT1, LUT2와 LUT_{MRC}에 $r_1, \ldots, r_n$이 입력되어야 한다.

$$T=\left|\sum_{i=1}^{n} a_{i,p}\right|_{M/M(p)},\quad T'=|M^{(p)}T|_{m_B}$$

$$b=\sum_{i=1}^{p} b_i,\quad b_i=|(M^{(p)}/m_i)|(M/M^{(p)})|(x_i/M_i)|_{m_i}|_{m_i}|_{m_B}$$



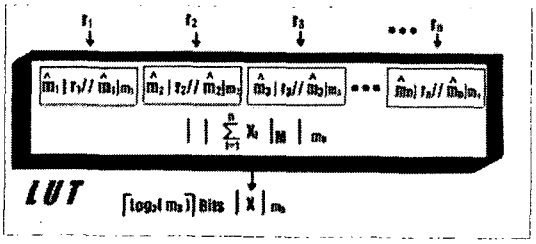
(그림 3) CRT를 사용한 기수 확장 구조
(Fig 3) Extension of Base using CRT

p=n−1인 경우 연산기의 크기가 각각 $(\prod_{i=1}^{n} m_i)\times\lceil Log_2(m_1-1)\rceil$ bits, $(\prod_{i=1}^{n} m_i)\times\lceil Log_2(m_B-1)\rceil$ bits이다. 또한, LUT3의 연산기의 크기는 $M_n\times(M-1)$ bits, LUT4는 $m_n\times\lceil Log_2(m_B-1)\rceil$ bits이고, LUT5는 $2\times m_B\times\lceil Log_2(m_B-1)\rceil$이다. LUT3 부분에서 조건2의 비교를 수행하려면, LUT_{MRC}에서 $|X_E|_M$을 산출하기 위한 MRC를 수행하여야 하므로, 1개의 LUT를 사용하는 경우는 연산기의 크기가 $(\prod_{i=1}^{n} m_i)\times\lceil Log_2(m_B-1)\rceil$ bits가된다. 그러므로 LUT연산기를 사용하여 복합연산을 수행하고, 연산속도 향상을 위하여 $|X_E|_M$을 산출을 1개의 LUT를 사용하면 4단계로 연산결과를 산출하지만, 연산기의 크기가 증대하는 문제점을 갖는다. 그러므로, 본 논문에서는 기존의 CRT연산법의 특성을 활용하여 모듈리에 따라 적합한 기수 확장기를 설계하고자 한다.

IV. 모듈리의 크기를 고려한 CRT를 사용한 개선된 기수확장

첫 번째, 모듈리의 크기가 작은 경우의 기수확장은 CRT의 기본방식에 LUT 연산기를 사용한다. 정수 x의 모듈리가 $m_1, \ldots, m_n$이고, 잉여수가 $r_1, \ldots, r_n$일 때, 모듈러스 m_B에 대한 기수확장을 수행하려면 식(8)의 CRT를 구하고 이 결과에 m_B의 잉여수를 구하면 된다. 그림 4와 같이 LUT를 사용하는 경우는 복합 연산이 가능하므로, 식(8)의 연산 및 m_B의 잉여수 생성과정은 한번에 수행할 수 있다. 기수확장은 1단계 이 후 출력되어 처리속도는 좋지만 연산기의 크기가 $(\prod_{i=1}^{n} m_i)\times\lceil Log_2(m_B-1)\rceil$이 되어 그림 3의 경우보다는 효율적이지만 모듈 리가 큰 경우보다는 다음과 같은 경우에 적용하는 것이 효과적이다. 예를 들어, 모듈리가 {3, 5, 7}일 때, $m_B=9$인 경우에 적용하면 LUT의 크기는

105×9 bits 임으로 효율적으로 사용할 수 있다.



(그림 4) LUT 연산기에 CRT를 사용한 기수확장

(Fig 4) Extension of Base using CRT with LUT

두 번째, 모듈리가 다수인 경우는 다음과 같이 수행한다. 식 (8)의 연산 시, 식(9)와 같은 연산 범위를 가지므로, 합산 시 수의 범위를 축소시키기 위하여, 식(14)와 같은 조건이 만족되는 경우이면, 연산기의 크기를 감소시킬 수 있다.

$$||X_N|_M|_{M_B} = ||X_N|_{M_B}|_M \text{ (단, } X_N < M) \quad (14)$$

모듈리가 3개인 경우, 식(8)에 따라 식(15-1)와 같이 표현하고, X_i간의 합을 A라 하면, 식(15-2)에서 X_N은 2M보다 적은 범위를 가지므로,

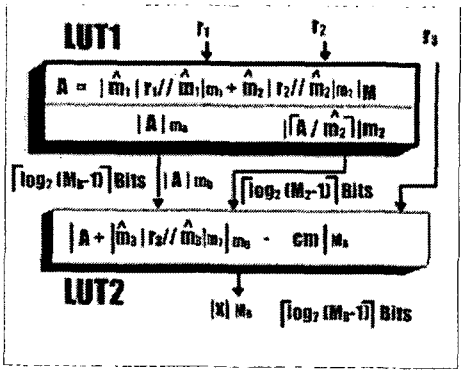
$$|X_N|_M = |X_1 + X_2 + X_3|_M, \text{ (N:정수)} \quad (15\text{-}1)$$

$$X_N = |A|_M + X_3 \quad (A = X_1 + X_2)$$

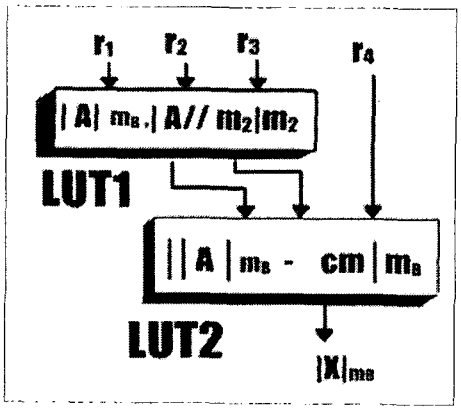
$$(0 \le |A|_M < M, \quad 0 \le X_3 < M - \overset{\wedge}{m_3},$$

$$0 \le X_N < 2M - \overset{\wedge}{m_3}) \quad (15\text{-}2)$$

식(14)의 조건을 만족시키기 위하여, 식(16-1)의 조건에서 c=1 이면, X_N이 M보다 크므로 식(15-2)에서 M을 감소하고, 식(16-2)와 같이 기수확장 모듈리 m_B를 개별적으로 적용할 수 있으므로, LUT를 사용하여 연산기를 구성할 때, 연산기의 크기를 감소시킬 수 있다.



(a)



(b)

(그림 5) 식(14)조건에 따른 CRT를 사용한 개선된 기수 확장 (a) 모듈리 3개인 경우 (b) 모듈리 4개인 경우

(Fig 5) The improved Extension of Base using CRT with the condition of equation(14)(NO. of moduli=3) (a) In case of No. of Moduli=3 (b) In case of No. of Moduli=4

$$c = \lceil X_N/M \rceil, cm = c*M \quad (16\text{-}1)$$

$$|X|_{m_R} = |X_N|_{m_R} = ||A|_{m_B} + |\overset{\wedge}{m_2}|_{r_2} \quad (16\text{-}2)$$
$$// \overset{\wedge}{m_2}|_{m_2}|_{m_R} - |cm|_{m_B}|_{m_R}$$

(그림 5)에 식(14)의 조건에 의한 식(16)의 기수확장을 그림 5. 나타내었으며, $|A|_{m_B}$에 대한 연산은 LUT1에서 수행하고, 식(16-1)의 조건에 대한 구분 값은 식(17-1)과 같이 압축된 비교 값 c_1은 LUT2로 전달되어, c_2와 식 (17-2)에 의해서 조건을 생성한다. 그러나, LUT를 사용하여 기수확장 연산기를 생성하므로, 기수 확장된 값은 식 (16-2)에 따라 LUT1과 LUT2에 선 저장되며, 조건생성은 식(17-3)과 같이 정수형으로 저장하여 수행한다.

$$c_1 = |A/\overset{\wedge}{m_2}|_{m_2},\ c_2 = |X_3/\overset{\wedge}{m_2}|_{m_2},$$
$$c_T = c_1 + c_2 \quad (17\text{-}1)$$

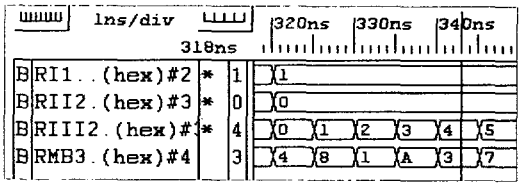
$$c = \lceil c_T/m_2 \rceil, cm = c*M \quad (17\text{-}2)$$

$$c_1 = |\lceil A/\overset{\wedge}{m_2} \rceil|_{m_2}, \quad (17\text{-}3)$$

모듈리가 4개인 경우는 $|A|_M = |X_1 + X_2 + X_3|_M$으로 하여 모듈 리가 3개인 경우와 동일한 방법으로 그림 5. (b)와 같이 수행할 수 있다. 그림 5 (a)인 경우, LUT1의 크기는 $r_1 \times r_2 \times (\lceil Log_2(m_B - 1) \rceil + \lceil Log_2(m_2 - 1) \rceil)$ 이고, LUT2의 크기는 $m_B \times m_2 \times m_B$이다.

V. 실험 및 고찰

본 논문에서 수행한 CRT를 사용한 기수확장에 대한 논리 검증은 HLL 및 VHDL을 사용하여 수행하였으며, 모듈리가 {3, 5, 7}이고, 기수확장용 모듈러스가 11인 경우 M=105이고, 생성된 결과를 그림 6에 표시하였다. 3개의 잉여수 입력은 RI1, RII2, RIII2로 표시하였으며, 기수확장된 출력 $|X|_{11}$은 RMB3로 나타내었다. $X = 25$인 경우, $r_1 = 1$, $r_2 = 0, r_3 = 4$이고, 기수 화장된 결과 $|X|_{11} = 3$를 16진수로 표시하였다.



(그림 6) CRT를 사용한 개선된 기수확장의 입출력 신호 (모듈리: {3, 5, 7}, 기수확장 모듈리 :11)

(Fig 6) Input-output signal of Improved Extension of Base using CRT(moduli: {3, 5, 7}, Extended Moduli: 11)

본 논문에서 설계된 기수 확장과 기존 확장방법과의 비교를 표 1에 표시하였다. LUT의 연산시간을 t_{LUT}라 하면, 연산속도는 그림 4의 경우가 가장 빠르지만, 모듈 리가 커지면 연산기의 크기가 증대함으로 모듈 리가 적은 경우에 사용하면 적합함을 알 수 있다. 또한, 그림 3의 경우는 연산속도는 모듈리에 관계없이 4 t_{LUT} 이지만, 연산기의 크기가 가장 크고, 그림 5의

경우는 모듈리가 큰 경우, 그림1 보다 연산기의 크기가 크지만 다른 연산기 보다는 연산기의 크기가 작으면서도 연산속도는 2 t_{LUT}로 향상되었음을 알 수 있다. 그러므로, 작은 모듈리의 경우는 그림 1의 기수 확장을 사용하고, 모듈리가 큰 경우에는 그림 5의 개선된 CRT를 사용한 기수확장을 사용하는 것이 적합함을 알 수 있다.

<표 1> 개선된 CRT응 사용한 기수확장의 연산시간 및 연산기 크기 비교

<Table 1> The comparison of operation time and LUT size of improved Extension of Base using CRT

기수 확장기 종류	연산표 크기(bits)				연산속도 (t_{LUT})	
	모듈리3,5,7		모듈리5,7,11,13			
	확장기수		확장기수		모듈리 수=3	모듈리 수=4
	4	11	3	9		
그림 1.	333	873	2045	3125	3	4
그림 3.	2146	3306	155213	165857	4	4
그림 4.	210	420	10010	20020	1	1
그림 5.	355	1645	3324	9164	2	2

VI. 결론

잉여수계를 사용한 연산기는 병렬성과 모듈리간의 자리올림수가 없는 장점을 가지므로, 디지털 신호처리 등과 같은 특수 목적의 프로세서 설계 시 적합하다. 그러나, 스케일링 등과 같이 적용되는 수의 범위를 확장하기 위하여 기수확장을 수행하여야 하는데, 기수확장 연산이 전체 시스템의 효율성을 감소시키는 요인이 된다.

그러므로, 본 논문에서는 기수확장 시 연산기의 크기 및 속도를 향시키기 위하여, 기존의 MRC를 사용한 방법과 CRT를 사용한 기수 확장을 고찰하여, 모듈리의 크기에 따라 적합한 기수확장을 수행하였으며, 모듈리가 작은 경우에는 CRT 연산을 LUT에 직접 적용한 기수확장을 사용하면 연산 속도를 향상시킬 수 있으며, 모듈리의 크기가 커지는 경우는 그림 5에 제안한 개선된 CRT를 사용한 기수 확장을 사용하는 것이 좋음을 알 수 있었다.

앞으로, 제안된 기수확장 법을 사용하여 대량의 모듈리에 대한 적용 시, 비교 정보의 압축 방법에 대한 연구가 더욱 계속되어야 할 것으로 사료된다.

참고문헌

[1] Nicholas S. Szabo, Richard I. Tanakas, (1967). Residue Arithmetic and its Applications to Computer Technology, *McGrow-Hill.*

[2] K.D.Weinmann, M.A. Soderstrand and S.Shebani, (1982), "Evaluation of New Hardware for a High-speed, Digital, Adaptive Filter Using the Residue Number System", *16th Asilomar Conference on Circuits, Systems, and Computers*, pp. 187-191.

[3] Michael A. Soderstrand, Bhaskar Sinha, (April, 1984),"A Pipelined Recursive Residue Number System Digital Filter", *IEEE Transactions on Circuits and Systems*, Vol. CAS-31, No. 4.

[4] 尹賢植, 趙源敬., (1993, 2月). "剩餘數係를 利用한 디지탈 神經網回路의 實現", 大韓電子工學會 論文誌, 第 30卷, B編, 第 2號, pp. 44-50.

[5] 金龍成 外 1, (1996, 1月). "高速 그래픽 處理를 위한 剩餘數係 乘算器 設計에 關한 硏究", 大韓電子工學會 論文誌, 第 33卷, B編, 第 1號, pp. 25-37.

[6] 金龍成, (1999, 8月), "MAC演算用 多重 剩餘數係에 대한 硏究", 驪州大學論文集 産業技術 硏究所, 第6集, pp.297-310.

[7] 金龍成, (1995, 11月), "多技能 高速 MRC設計에 關한 硏究", 驪州大學 産業技術 硏究所 論文集, 第1卷, pp.317-332.

[8] Kenneth H. O'Keefe ,(November.1975)," A Note on Fast Base Extension for Residue Number Systems with Three Moduli", *IEEE Trans. Computer*, Vol C-24, pp1132-1133.

[9] Ferruccio Barsi and M. Cristina Pinotti, (October, 1995), "Fast Base Extension and Precise Scaling in RNS for Look-Up Table Implementation", *IEEE Trans. on Signal Processing, Vol 43, No. 10.*

A Study on the Extension of Base Using CRT in RNS

Yong-Sung, Kim*

Abstract

The Extension of Base is a fundamental Method to expend the moduli in RNS(Residue Number System). RNS has the benefit of parallelism and no carry propagation at each moduli, but division , extension of base and etc. is the problem of RNS in case of the operation speed.Generally this method is applied to system using Mixed Radix Conversion. it appears to decrease the size of Arithmetic Unit, but increasing the time of operation.

So in this paper, the Improved Extension of Base is proposed using Chinese Remainder Theorem. it has the comparative small size and Improved speed.