韓國 컴퓨터情報學會 論文誌
第12卷 第3號, 2007. 7.

2007-12-3-1-5

생산자동화 시스템에서 실시간 물체인식을 위한 디지털 뉴런프로세서의 설계 및 구현

홍 봉 화*, 주 해 종**

Design and Implementation of the Digital Neuron Processor for the real time object recognition in the making Automatic system

Bong-Wha Hong*, Hae-Jong Joo**

요 약

본 논문에서는 캐리전파가 없어 고속연산이 가능한 잉여 수 체계(Residue Number System)를 이용하여 생산자동화 시스템에서 실시간 물체인식을 위한 고속의 디지털 뉴런 프로세서를 제안하고 이를 구현하기 위한 중요연산부인 PE를 설계 및 구현하였다. 설계된 디지털 뉴런프로세서는 잉여수계를 이용한 MAC(Multiplier and Accumulator)연산기와 혼합계수 변환을 이용한 시그모이드 함수 연산부로 구성된다. 설계된 회로는 C언어 및 VHDL로 기술하였고 Compass툴로 합성하였으며 LG 0.8μm CMOS공정으로 설계되었다. 실험결과, 본 논문에서 설계 및 구현한 디지털 뉴런프로세서는 기존 방식의 잉여수계를 이용한 연산기 및 실수연산기로 구현한 뉴런프로세서에 비하여 3배 이상의 연산속도와 약 50%정도 하드웨어 크기를 줄일 수 있었다. 본 논문에서 설계 및 구현한 디지털 뉴런프로세서는 실시간 처리를 요하는 생산자동화 시스템의 물체인식 시스템에 적용될 수 있을 것으로 기대된다.

Abstract

In this paper, we designed and implementation of the high speed neuron processor for real time object recognition in the making automatic system. and we designed of the PE(Processing Element) used residue number system without carry propagation for the high speed operation. Consisting of MAC(Multiplication and Accumulation) operator using residue number system and sigmoid function operator unit using MRC(Mixed Radix conversion) is designed.

The designed circuits are descript by C language and VHDL(Very High Speed Integrated Circuit Hardware Description Language) and synthesized by compass tools and finally, the designed processor

• 제1저자 : 홍봉화
• 접수일 : 2007.5.31, 심사일 : 2007.6.8, 심사완료일 : 2007. 7.20.
*경희사이버대학교 정보통신학과 교수 **인덕대학 산학협력단 교수
※ 이 논문은 2005년도 정부의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-2005-003-D00182)

is fabricated in 0.8μm CMOS process. we designed of MAC operation unit and sigmoid proceeding unit are proved that it could run time 0.6nsec on the simulation and improved to the speed of the three times and decreased to hardware size about 50%, each order. The designed neuron processor can be implemented of the object recognition in making automatic system with desired real time processing.

▸ Keyword : 신경회로망(Neural Network), 역전파 알고리즘(Back propagation Algorism), 잉여수 체계(Residue Number System), 혼합기수변환(Mixed Radix Conversion), MAC(Multiplication and Accumulation)

Ⅰ. 서 론

신경회로망 이론은 1940년대에 이미 제안되었지만 반도체 기술이 발전한 근래에 이르러 많은 발전을 하게 되었다. 현재까지 연구되고 있는 신경회로망의 응용분야는 인식(패턴 인식, 음성인식), 제어이론, 영상처리분야 등이 있다[1]-[5][15][16].

최근 영상신호처리 및 패턴인식 분야에서 대량의 데이터를 실시간으로 처리 하여야하는 필요성이 증가하고 있다. 컴퓨터와 사용자간의 인터페이스 문제에 있어서 실시간 처리는 중요한 해결 수단이다. 이와 같은 응용분야에 신경회로망을 이용하기 위해서는 대량의 데이터를 실시간으로 처리할 수 있는 고속의 MAC연산기와 판별함수를 고속으로 처리할 수 있는 연산기가 요구된다.

신경회로망의 연산은 행렬·벡터 연산을 기본으로 하고 있으므로 어레이프로세서의 구조로 구현이 가능하다. 또한, 잉여 수 체계는 모듈러간의 캐리 정보가 필요 없으며, 승산이 가산과 거의 같은 속도로 구현될 수 있으므로 고속 MAC 연산기의 구현이 가능하다[6]-[13].

본 본문에서는 잉여수계를 적용하여 고속의 MAC 연산기를 설계하고 이 설계된 MAC 연산기를 이용하여 고속의 디지털 뉴런프로세서를 설계 및 구현하고자 한다.

뉴런프로세서의 구현 시 문제가 되는 시그모이드 함수처리는 잉여수계 (Residue Number System)를 이용한 ROM 테이블(ROM Look-Up Table)방식을 사용함으로써 효율적으로 수행할 수 있다[9]-[11].

신경회로망의 기능을 디지털회로를 이용한 어레이프로세서로 구현하는 것은 아날로그 방식 및 광을 이용한 신경회로망이 아직 연구단계에 있는 것에 비하여 현재의 발전된 디지털 VLSI 설계 기술을 이용하여 실제로 이용 가능한 수준의 신경 칩을 제작할 수 있기 때문에 중요하다.

잉여 수 체계를 이용한 MAC 연산기의 설계 시 가산과 승산이 동일한 동형으로 연산 하기위하여 순환 군을 사용하며, 연산기에 사용되는 순환 부호는 각각의 부호에 한 비트만을 "1"로 하는 코드를 사용함으로써 처리 시간을 줄일 수 있고, 하드웨어를 단순하게 구현 할 수 있다.

설계된 MAC 연산기는 200MHZ 이상의 속도로 MAC연산(AC <-AC ± A * W)을 수 행 할 수 있으며, 이 MAC 연산기를 내장한 PE를 어레이 형태로 나열함으로써 대 규모의 신경회로망 구현이 가능하다. 또한, 하나의 칩에 64개 정도의 PE(Processing Element)를 내장시킴으로써 신경회로망 이론을 실제 응용하는 제품개발이나 신경회로망의 이론적인 연구를 위하여 공헌할 수 있을 것으로 기대된다.

Ⅱ. 역전파 신경회로망

역전파(Back propagation) 알고리즘은 1986년경 Rumelhart로 대표되는 PDP그룹에 의해 제한되었으며, 그림 1과 같은 구조로 되어있다. 각 층이 N_L개의 노드를 가지고 L층으로 구성된 역전파 신경회로망을 고려하자.

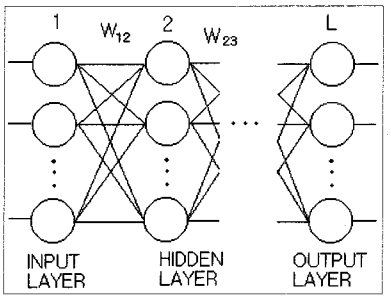


그림 1. 역전파 신경회로망
Fig. 1. Back propagation neural network

그림 1에서 j층의 한 노드에 대한 입력(net_j)과 활성함수를 통과한 후의 출력은 다음식과 같다.

$$net_j = \sum W_{ji} O_i = U_j \quad \cdots\cdots (1)$$

$$O_j = F(U_j) = O_j = \frac{1}{1+\exp(-(net_j+\theta_j)/\theta_o)} \quad \cdots\cdots (2)$$

(θj : 문턱값, θo : 기울기)

시그모이드 활성함수는 출력 값을 0과 1사이의 값으로 제한하며, 다음 단 노드의 입력 값이 된다. 단일노드의 연산은 층의 순서에 의해 식 (1)과 (2)의 연산을 반복 수행하며, 최종 층 k층에서의 출력 O_k를 출력한다. 여기서 목표 값을 T_k라 하고, 실제 출력 값을 O_k라 하면, 오차 값 E는 식(3)의 최소자승오차(LMS : least mean square) 식으로써 구해질 수 있다[1]-[5].

$$E_p = \frac{1}{2}\sum (T_k - O_k)^2 \quad \cdots\cdots (3)$$

실제출력과 훈련(training)패턴간의 학습과정은 연결강도의 변경에 의해 처리된다. 연결강도 변경은 식 (3)에 의해 얻어진 오차 E에 의해서 조절된다. 그림 1에서 j층의 어느 한 노드의 p번째 패턴의 오차 값 δ_{pj}는 식 (4)와 같다.

$$\delta_{pj} = O_{pj}(1-O_{pj})\sigma_{pj} \quad \cdots\cdots (4)$$

여기서 상위층인 k층의 오차 값에 의하여 σ_{pj}는 식 (5)처럼 표현된다.

$$\sigma_{pj} = \sum \delta_{pk} W_{kj} \quad \cdots\cdots (5)$$

최종 층 k에서 p번째 패턴 오차 σ_{pk}는 식 (6)에 의해서 구할 수 있다.

$$\sigma_{pk} = T_{pk} - O_{pk} \quad \cdots\cdots (6)$$

i 층과 j층 사이의 t+1단계 연결강도 변화량 $\triangle W_{ji}(t+1)$은 식 (7)과 같다.

$$\triangle W_{ji}(t+1) = \eta(\delta_j O_i) + \alpha \triangle W_{ji}(t)$$

$$= \eta(\delta_j O_i) + \alpha \triangle W_{ji}(t) \quad \cdots\cdots (7)$$

여기에서 η와 α는 각각 학습계수와 관성계수로서 1회의 연결강도 갱신에 따른 비례 값이다. 각 층에서 계산된 오차는 역방향으로 연결강도를 순차적으로 갱신한다.

III. 디지털 뉴런프로세서의 설계

본 연구에서는 아날로그 신경회로망의 연산방식을 분석하고 디지털회로 구현 시 문제가 되는 점에 대하여 분석하였다. 또한, 다량의 데이터를 실시간으로 처리할 수 있는 고속 디지털 뉴런 프로세서를 설계 및 구현하고, 구현된 뉴런프로세서를 생산자동화 시스템에서 실시간 물체인식에 응용함을 목적으로 하며, 그림 2와 같은 설계단계를 가진다.

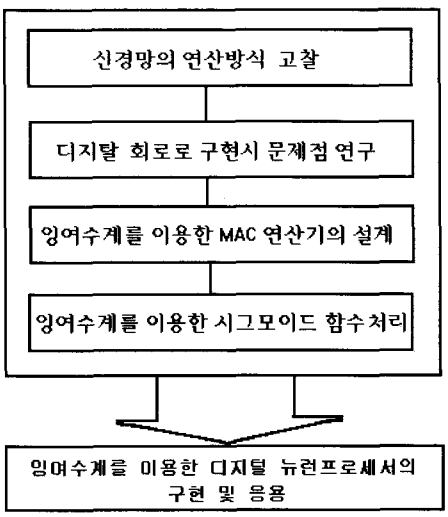


그림 2. 디지털 신경회로망의 설계단계
Fig. 2. Processing of the design of the digital neural networks

1. 신경회로망의 기능을 어레이프로세서로 설계

II절의 식들은 행렬식을 이용하여 다음과 같이 표현이 가능하다.

$$U = {}^t[U_1 \cdots U_n]$$
$$O = {}^t[O_1 \cdots O_n]$$
$$\delta = {}^t[\delta_1 \cdots \delta_n]$$
$$T = {}^t[T_1 \cdots T_n]$$

$$W = \begin{pmatrix} W_{11} & \cdots & W_{1n} \\ & \cdots & \\ & \cdots & \\ & \cdots & \\ W_{n1} & \cdots & W_{nn} \end{pmatrix}$$

$U = W \cdot O$

$O = F(U)$ 이다.

윗 식들은 그림 3과 같은 구조의 어레이프로세서 기법에 의하여 처리될 수 있다.

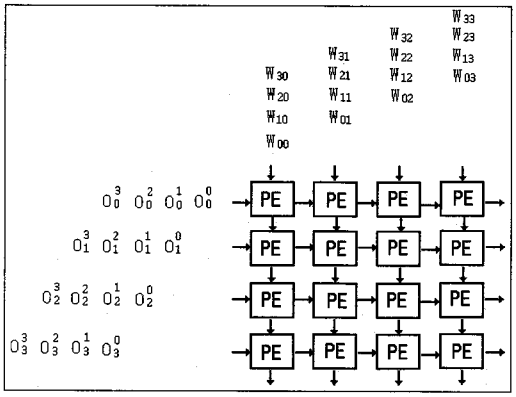


그림 3. 그림 3. 신경망 모델의 2차원 어레이프로세서 구현
Fig. 3. Implementation of two-dimensional Array Processor for neural network

그러나 그림 3과 같이 2차원 어레이 구조를 하드웨어로 구현 시, 속도는 빠르지만 하드웨어 크기가 증대되는 문제가 발생한다.

그림 3의 구조는 어레이프로세서 설계기술에 의하여 그림 4와 같이 2차원 구조를 1차원으로 정합하는 것이 가능하며, 이러한 1차원 구조는 대량의 데이터 처리 시 분할처리도 가능하기 때문에 본 연구에서는 그림 4의 구조로 설계한다.

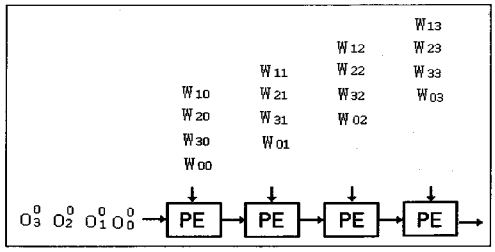


그림 4. 역 전파 신경회로망의 1차원 어레이프로세서로의 구현
Fig. 4. Implementation of one-dimensional Array processor for neural networks

그림 4에서 각 PE의 기본연산은 식 (8)과 같은 MAC연산이다.

AC ⇦ AC ± A * W ..... (8)

식 (8)과 같은 연산을 수행하는 MAC의 기본 구조는 그림 5와 같다.

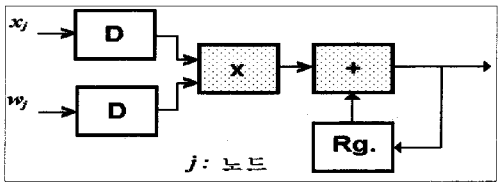


그림 5. MAC의 기본구조
Fig. 5. Structure of MAC

그러나, 식 (8)을 현재의 연산회로(정수형과 부동 소수점 연산회로)를 가지고 구현할 경우, 하드웨어 크기의 증대 및 속도가 저하되는 문제점이 발생됨으로 새로운 방식의 MAC 연산회로의 설계와 이를 이용한 어레이프로세서의 설계 방식이 연구되어야 한다.

2. 잉여수계를 이용한 설계

2.1. 정수의 잉여수 표현

일반적인 2진수 체계와는 달리 가중치(Weight)가 없는 수체계로서 서로소(Relatively Prime)로 이루어진 모듈리(Moduli)를 이용하며, 각 모듈러스(Modulus)로 나눈 나머지만으로 수를 표현한다. 연산시 자리올림이 없으므로 연산이 각 모듈러스 단위로 독립적으로 동작한다. 그러므로 연산속도의 고속화가 가능하며, 승산도 가산과 동일한 구조의

연산표를 이용하여 가산과 동일한 속도로 실행할 수 있기 때문에 고속의 승산이 많이 필요한 디지털 신호처리, 영상처리등과 같은 실시간 처리를 요하는 분야에 자주 사용되고 있다[6]-[11][14]. 신경회로망의 노드연산은 행렬과 벡터의 승산과 누적으로 표현할 수 있으며, 사용되는 수의 범위가 크지 않고, 각 노드의 출력은 일정한 수의 범위로 제한되기 때문에 잉여수계를 이용하여 효과적으로 구현할 수 있다. 잉여수계서 모듈리를 (m_1, m_2, ..., m_N)으로 취할 경우, 임의의 정수 X는 식 (9)와 같이 몫 Q_i와 잉여수 $|X|_{mi}$로 표기할 수 있다.

$$X = Q_i + |X|_{mi} \text{(단 } 0 \leq |X|_{mi} < m_i \text{이며, } i = 1, 2, ..., N) \cdots\cdots (9)$$

여기서 $|X|_{mi} = X \bmod m_i$로 한다.

식 (9)를 이용하여 식 (10)과 같이 N차원 잉여수로 표시할 수 있다.

$$X \Leftrightarrow \{|X|_{m1}, |X|_{m2}, ..., |X|_{mN}\} \cdots\cdots (10)$$

이 경우 식 (10)로 표현할 수 있는 정수 X의 범위는 식 (11)과 같다.

$$0 \leq X \leq M\left(M = m_1 m_2 \cdots m_N = \prod_{i=1}^{N} m_i\right) \cdots\cdots (11)$$

잉여수계에서는 식 (11)과 같이 정수만을 취급하고 표현할 수의 범위가 각 모듈리의 곱으로 제한된다.

예를 들어 모듈 리가 3개인 경우, m_1=2, m_1=3, m_1=5일 때 -4에서 33까지의 정수에 대한 잉여수 표현을 표 1에 나타내었다[6]. 정수 0에서 29까지의 잉여수는 서로 독립적으로 표현되지만, 0보다 작거나, 30이상인 정수의 경우는 정수 0과 29까지를 표현한 잉여수가 반복사용 되므로 잉여수에 의하여 표시될 수 있는 수의 범위는 식 (11)과 같이 $0 \leq X < 30$이 된다. 그러므로 잉여수에 의하여 표현되는 정수의 범위를 증가시키기 위해서는 모듈리의 수를 증가시키거나, 각 모듈러스의 값을 증가시켜야 한다.

표 1. 정수 -4에서부터 33까지의 잉여수 표현
Table 1. Residue representation of the number -4 to 32

정수 X	잉여수 표현 m1 m2 m3	정수 X	잉여수 표현 m1 m2 m3
-4	0 2 1	15	1 0 0
-3	1 0 2	16	0 1 1
-2	0 1 3	17	1 2 2
-1	1 2 4	18	0 0 3
0	0 0 0	19	1 1 4
1	1 1 1	20	0 2 0
2	0 2 2	21	1 0 1
3	1 0 3	22	0 1 2
4	0 1 4	23	1 2 3
5	1 2 0	24	0 0 4
6	0 0 1	25	1 1 0
7	1 1 2	26	0 2 1
8	0 2 3	27	1 0 2
9	1 0 4	28	0 1 3
10	0 1 0	29	1 2 4
11	1 2 1	30	0 0 0
12	0 0 2	31	1 1 1
13	1 1 3	32	0 2 2
14	0 2 4	33	1 0 3

($m_1 = 2$, $m_2 = 3$, $m_3 = 5$, $M = 30$)

2.2 잉여수의 기본연산

표 1에서 정수 X=25는 $|X|_{m1}$=1, $|X|_{m2}$=1, $|X|_{m3}$=0 즉(1, 1, 0)로 표현되며 기본 연산은 식 (12)와 같다.

$$Z = X \circ Y \ (\text{'}\circ\text{'} : *, +, -) \cdots\cdots (12)$$

또한, 식 (12)는 식 (13)과 같이 나타낼 수 있다[6].

$$|Z|_{mi} = |X|_{mi} \circ |Y|_{mi} = |X \circ Y|_{mi} \cdots\cdots (13)$$

예를 들면 X = 12, Y = 5의 가산의 경우, 다음과 같이 계산된다.

	mod 2	mod 3	mod 5	
	0	0	2	; x = 12
+	1	2	0	; y = 5
	1	2	2	; Z = 17

연산 결과는 표 1을 참조하면 17임을 알 수 있다.

2.3. 혼합기수 변환(Mixed Radix Conversion)

모듈리를 m_1, m_2, ..., m_N으로 선택한 경우, 정수 X의 동적 구간은 $[0, \prod_{i=1}^{N} m_i - 1]$이며, 여가서 X의 잉여수는 식 (14)와 같이 표현될 수 있다.

$$X \Leftrightarrow \{X|_{m1}, X|_{m2}, ..., X|_{mN}\} = \{ r_1, r_1, ..., r_N \} \quad \cdots\cdots (14)$$

또한, 식 (15)와 같이 혼합기수(Mixed Radix)로 표현할 수 있다[6]-[8].

$$X = a_N \prod_{i=1}^{N-1} m_i + ,... + a_3 m_1 m_2 + a_2 m_1 + a_1 \quad \cdots\cdots (15)$$

식 (15)에서 a_i는 i번째 자리수이며, $0 \le a_i < m_i$이다.

혼합기수로 변환 시 혼합기수의 계수를 구하는 과정은 아래와 같이 구할 수 있다.

i) 식 (15)의 양변에 모듈로 m_1을 취하면 마지막 항을 제외하고는 모두 m_1의 곱으로 되어 있으므로 식 (16)과 같이 기술된다.

$$| X |_{m_1} = a_1 \quad \cdots\cdots (16)$$

식 (16)에서 a_1은 첫 번째 잉여수인 r_1이 된다.

ii) 식 (15)의 양변에 a_1을 빼주고 모듈로 m_2를 취하면 식 (17)과 같다.

$$|X - a_1|_{m2} = \left| a_N \prod_{i=1}^{N-1} m_i + \cdots + a_3 m_1 m_2 + a_2 m_1 \right|_{m2} \quad \cdots\cdots (17)$$

$$= |a_2 m_1|_{m2}$$

$$a_2 = \left| \frac{x - a_1}{m_1} \right|_{m2}$$

$$= \left| \left[\frac{X}{m_1} \right] \right|_{m2}$$

iii) i), ii)의 방식으로 나머지 혼합기수 계수를 구하면 식 (18)과 같다.

$$a_i = |[\frac{X}{m_1 m_2, \ldots, m_{i-1}}]| \quad \cdots\cdots (18)$$

식 (18)에서 i 〉 1 일 경우 계수 a_i는 식 (19)과 같이 표현된다.

$$a_i = |[\frac{x}{m_1 m_2 \cdots m_{i-1}}]|_{m_i} \quad \cdots\cdots (19)$$

2.4. 잉여수계를 이용한 MAC의 기본구조

신경회로망의 기본연산은 행렬연산으로 표현되며, 이와 같은 연산은 식 (8)과 같이 승산과 가산이 혼합된 누산(MAC :Multiplier with Accumulator) 연산을 기본구조로 한다. 모듈리를 m_1,m_2,m_3이라 할 때, 두 입력 X, W의 누적 연산을 하는 잉여수 연산기의 기본 구조를 그림 6에 나타내었다.

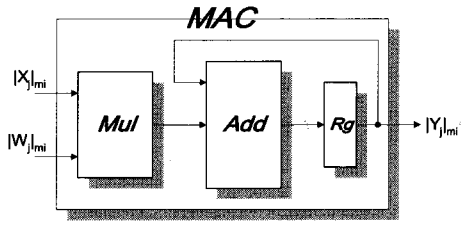


그림 6. 잉여수계를 이용한 MAC 연산기의 구성
Fig. 6. The structure of Mac Using residue number system

3. 순환군을 이용한 디지털 뉴런프로세서의 설계

3.1 순환 군

집합 G가 공집합이 아니고 정수 a,b에 대하여 a,b∈G이고, 연산 ' ○'가 a○b ∈G일 때, 이 연산이 결합법칙, 교환법칙이 성립하고 항등원 및 역원이 존재하면, 연산 ' ○'에 관하여 집합 G는 군을 이루며, <G, ○>로 표기한다. 또한, <G, ○>에 대하여 g∈G라 할 때 생성원(generator) g에 의해서 생성된 순 환 부분군 <g>는 식 (20)과 같으며, G=<g>인 순환군이라 한다[6]-[8].

$$\{ g^n \mid n \in Z \} = \{ ...,g^2,g^1,g^0,g^{-1},g^{-2},... \} \quad \cdots\cdots (20)$$

(g : 생성원, g^0 : 항등원, Z,g,n : 정수)

식 (20)에서 순환군 G가 위수n의 유한군이면 $G=\{g^0,g^1,g^2,...g^{n-1}\}$이고 $g^n = g^0$이다. 모듈러 연산은 순환군을 형성하므로 모듈러 연산기는 군론을 기초로 하여 설계될 수 있다. 연산 '×'에 대한 n차 순환군 {G:×}의 연산은 표 2와 같다.

표 2. n차 순환 군
Table 2. Oder n cycle group

×	g^0 g^1 g^2 . . . g^{n-1}
g^0	g^0 g^1 g^2 . . . . . g^{n-1}
g^1	g^1 g^2 g^3 . . . . . g^0
g^2	g^2 g^3 g^4 . . . . . g^1
.	. .
.	. .
.	. .
g^{n-1}	g^{n-1} g^0 g^1 . . . . . g^{n-2}

순환 군 {G:×}의 정의로 부터 mod m가산은 m차 순환 군이고, 임의의 mod p승산은 (p는 소수, 0디지트는 제외) (p-1)차 순환 군이다. 일반적으로 mod p승산에 대하여, 생성원은 Fermat의 이론으로부터 다음과 같이 구 할 수 있다 [6]-[8]. $|g^{p-1}|_p$가 가장 작은 g^{p-1} mod p의 잉여수일 때, g가 조건 $|g^{p-1}|_p = 1$을 만족한다면, 정수 $g < p$는 mod p-1승산의 생성원이다. 예를 들어, 발생기가 2인 경우, mod 11의 승산은 10차 순환 군이고, 표 3(b)와 같이 순환 군을 형성하기 때문에 모듈러 연산에 이용될 수 있다. 임의의 순환 군에 대해서 p가 소수인 경우 mod p-1 승산(0 디지트 제외)에 대한 순환군은 mod (p) 가산의 순환 군과 1 대 1 대응하여 사상(mapping)될 수 있으므로 동형(isomorphic)이다.

표 3. mod 11 가산과 mod 10의 동형관계
(a) mod 11 가산, (b) mod 10 승산(g=2)
Table 3. Isomorphic mod 11 addition and mod 10 multiplication
(a) mod 11 addition, (b) mod 10 multiplication(g=2)
(a)

+	0	1	2	3	4	5	6	7	8	9	10
0	0	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10	0
2	2	3	4	5	6	7	8	9	10	0	1
3	3	4	5	6	7	8	9	10	0	1	2
4	4	5	6	7	8	9	10	0	1	2	3
5	5	6	7	8	9	10	0	1	2	3	4
6	6	7	8	9	10	0	1	2	3	4	5
7	7	8	9	10	0	1	2	3	4	5	6
8	8	9	10	0	1	2	3	4	5	6	7
9	9	10	0	1	2	3	4	5	6	7	8
10	10	0	1	2	3	4	5	6	7	8	9

(b)

×	1	2	4	8	5	10	9	7	3	6
1	1	2	4	8	5	10	9	7	3	6
2	2	4	8	5	10	9	7	3	6	1
4	4	8	5	10	9	7	3	6	1	2
8	8	5	10	9	7	3	6	1	2	4
5	5	10	9	7	3	6	1	2	4	8
10	10	9	7	3	6	1	2	4	8	5
9	9	7	3	6	1	2	4	8	5	10
7	7	3	6	1	2	4	8	5	10	9
3	3	6	1	2	4	8	5	10	9	7
6	6	1	2	4	8	5	10	9	7	3

3.2. 순환 군을 이용한 연산

표 3에 나타낸 순환 군을 이용하여 연산기의 논리회로를 설계할 경우 순환 군을 2진 부호화하여야 한다. 순환 군을 2진 부호화하는 기본적인 방법은 순환군과 1 대 1 대응된 2진 부호화를 수행하여 논리회로로 설계하는 것이다. 표 2는 mod n 승산의 순환군인 경우 기본요소의 수가 n개 이고 mod p의 승산을 수행하므로 p=n이며, g^m은 $0<g^m\leq(p-1)(0<m\leq n-1)$이므로 각 기본 요소 당 소요되는 비트수 b는 $b=[\log_2(n-1)]$이다. 그러므로 순환군의 각 요소를 1 대 1 사상하여 2진 부호화 하는 경우 1개의 연산기에 $n^2\times[\log_2(n-1)]$ 이상의 논리연산이 필요하므로 연산기의 크기를 감소시키기 위하여 표 4와 같이 1-out-of-m 코드 방식을 이용하였다.

표 4. 1-out-ot-10 코드 구성
Table 4. The structure of 1-out-of-10 code

십진수	1-out-of-10									
0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	1	0	0	0
4	0	0	0	0	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0
6	0	0	0	1	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0

표 4에서 알 수 있듯이 1-out-of-m코드 방식은 m개의 비트 중 한 개의 비트만 고유 값을 나타내는 코드방식으로써, 잉여수계를 이용한 연산회로와 같이 순환 군을 이루는 경우에 효율적으로 적용될 수 있다.

IV. 잉여수계를 이용한 디지털 신경회로망의 설계

잉여수계 연산은 입력된 수를 선택된 모듈리에 따라 분할하고 각 모듈리 별로 연산하므로, 연산기의 크기가 감소되며 모듈리 간에 캐리정보가 필요 없으므로 고속의 연산을 수행할 수 있다. Ⅲ절의 표 3과 같은 순환 군과 표4의 1-out-of-m 코드방식의 부호화 기법을 이용 하여 고속의 잉여 수 연산기를 설계할 수 있으며, 이 설계된 연산기를 뉴런프로세서의 핵심부분인 MAC연산부에 적용하여 그림 7과 같은 고속의 디지털 뉴런프로세서를 구현하고자 한다.

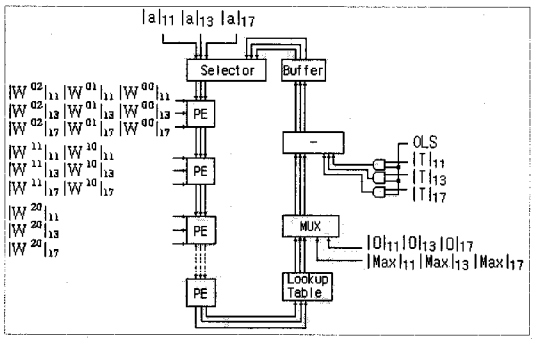


그림 7. 잉여수계를 이용한 뉴런 프로세서의 구성
Fig. 7. Configuration of neuron processor using RNS

그림 7의 뉴런프로세서는 결합강도와 입력과의 승산 및 누산을 수행하는 MAC연산부와 MAC연산부의 처리결과를 입력으로 하는 시그모이드 함수연산부로 구성되며, 각 PE의 블록도는 그림 8과 같다..

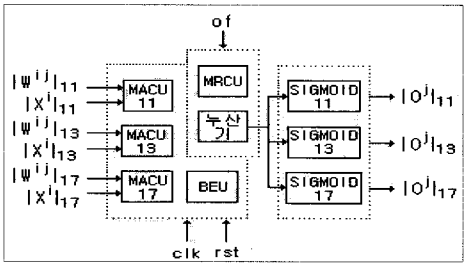


그림 8. 잉여수 체계를 이용한 역전파 신경회로망의 PE블록도
Fig. 8. The block of PE for back propagation neural networks using residue number system

1. 고속 MAC 연산회로의 설계

잉여수계를 이용한 MAC연산기의 설계 시 가산과 승산이 동일한 속도로 수 행 된다. 잉여수계를 이용한 가산과 승산은 표 3 과 같이 순환 군을 형성하기 때문에 그림 9와 같은 바렐쉬프터(barrel shifter)의 기본구조를 사용하여 고속의 연산회로를 설계할 수 있다.

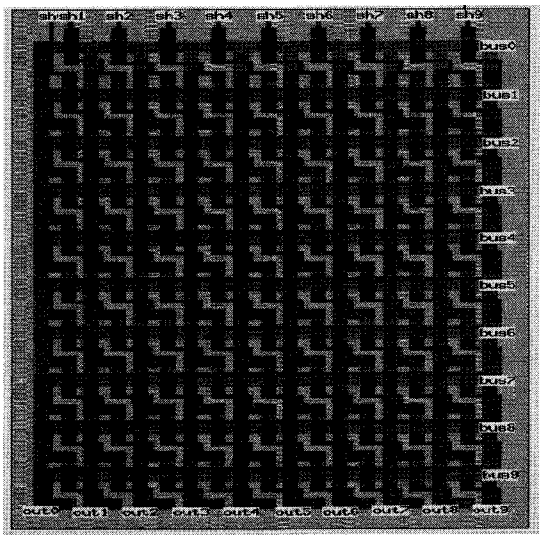


그림 9. 바렐쉬프터의 기본형태 (10×10)
Fig. 9. Basic configuration of barrel shifter(10×10)

그림 9에서 sh_i는 입력, bus_i는 선택, out_i는 출력을 나타내며, 선택선 bus_i에 따라, 입력된 데이터의 i비트 순환된 결과를 출력하며, i번 순환된 속도가 동일하므로, 바렐쉬프터를 이용하여 순환부호용 연산기를 구현할 수 있다. 바렐쉬프터를 이용한 모듈러 p의 가산기와 p-1 승산기를 그림 10과 11에 나타내었다.

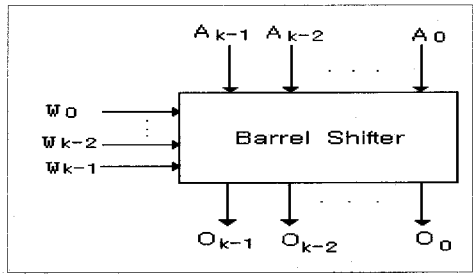


그림 10. 바렐 쉬프터를 이용한 가산기
Fig. 10. The Structure of Adder using barrel shifter

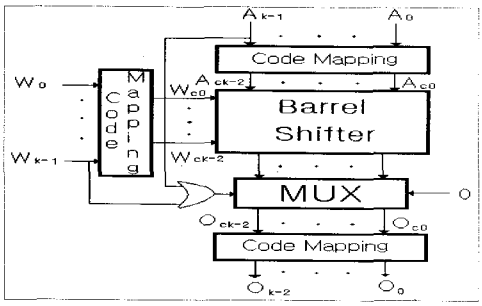


그림 11. 바렐쉬프터를 이용한 승산기의 구조
Fig. 11. The Structure of multiplier using barrel shifter

그러나, 모듈러 (p-1)의 승산기는 모듈러 p 의 가산기를 이용하여 설계하므로 승산기의 입력 및 출력 시, 표 3의 동형관계에 따라 부호 정합용 해독기가 필요하며, 승산기의 입력 '0' (m_0=0)인 경우는 출력이 '0'이 되도록 한다.

본 논문에서는 부호정합용 해독기를 사용하지 않기 위하여 표 4의 1-out-of-m 코드 방식을 이용하고자 한다. 이 방식은 m 비트 중 한 비트 만 고유 값을 가지므로 부호 정합 시 각 비트가 1 대 1 대응되므로 입·출력 선을 재배열함으로써 부호정합이 가능하며, 부호를 재생성하기 위한 부가회로가 필요하지 않다. 그림 12는 위에서 기술된 내용을 기본 하여 구현된 MAC 연산기를 나타낸다.

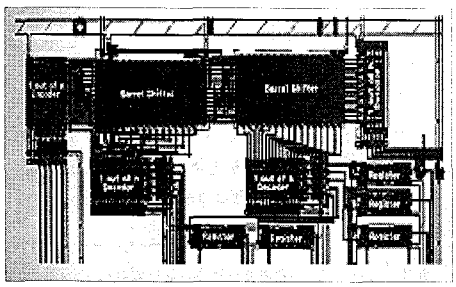


그림 12. MAC 연산부의 레이아웃
Fig. 12. The layout of MAC operation unit

그림 12의 MAC 연산부는 각 모듈 리가 11, 13, 17인 입력 벡터와 연결강도의 행렬의 곱 연산을 수행한 후 누적되며, 승산과 가산 모두 1 clock으로 연산을 수행할 수 있다.

2. 잉여수계를 이용한 시그모이드 함수부의 처리

시그모이드 함수처리는 II절의 식(2)와 같이 표현되며 이 부분은 신경회로망의 디지털회로 구현 시 문제가 되는 부분이다. 본 연구에서는 이 부분을 잉여수계 의 MRC(Mixed Radix Conversion)을 사용하여 구간을 분할하여 처리하고자 한다. 식(2) 에서 θ_j,θ_0에 따라 시그모이드 함수는 그림 13과 같은 특성을 갖는다.

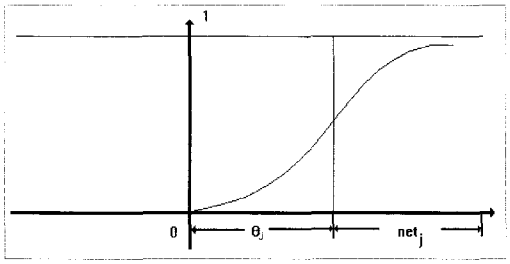


그림 13. θj 와 θ0에 의한 시그모이드 함수의 이동과 기울기
Fig. 13. Shifting and Slope of sigmoid function by θj and θ0

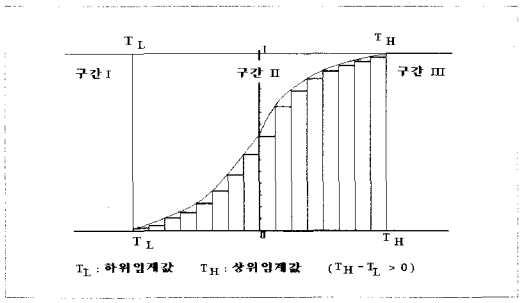


그림 14. 시그모이드 함수의 구간설정
Fig. 14. Decision of boundary in sigmoid function

그림 13를 그림 14와 같이 3구간으로 분할하여 처리하고, 구간 분할시 MRC를 사용하였다. 시그모이드 함수 처리부분을 그림 15와 같이 구성하였다.

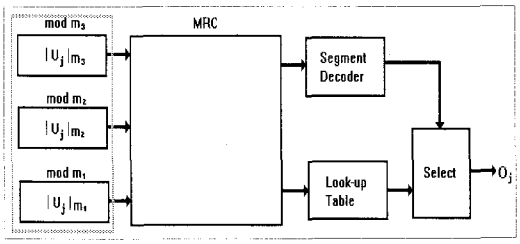


그림 15. 시그모이드 함수의 연산처리과정
Fig.15. Processing of sigmoid function

그림 15에서 MRC처리부분을 나타내면 그림 16과 같으며, 내부연산과정은 식 (17)에 의하여 이루어진다.

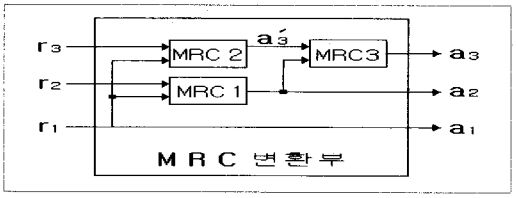


그림 16. 혼합 기수변환(MRC) 처리부분
Fig. 16. Processing unit of Mixed radix conversion

그림 16에서 잉여수계 입력(r_1,r_2,r_3)를 받아서 MRC(혼합 기수 변환)를 거쳐 가중 치 체계의 수로 변형된 출력(a_1,a_2,a_3)은 연산 표(Look up table)에 저장된 잉여 수를 찾아 출력한다. 가중치 체계로 변환된 출력 a_3는 연산표의 입력 값으로 구간 설정에 쓰이며, a_1, a_2를 이용하여 그림 14에 있는 구간 Ⅱ에 해당되는 값을 사전에 저장된 연산표의 값으로 출력한다.

3. 잉여수계를 이용한 디지털 신경회로망의 설계

각 모듈리 m_i에 따라 입력 값(X_i)과 결합강도(W_j)의 승산 및 누적 연산이 MAC연산부에 의해서 모듈리 별로 수행되며, 연산결과가 시그모이드 함수처리부에 입력되어 MRC연산을 거쳐 연산표에 의하여 표본화된 시그모이드 함수의 출력인 결과 출력 값(O_j)이 생성된다. 모듈리 m_1, m_2, m_3를 갖는 디지털 신경회로망을 잉여 수 체계를 이용하여 구성할 경우 그림 17과 같다.

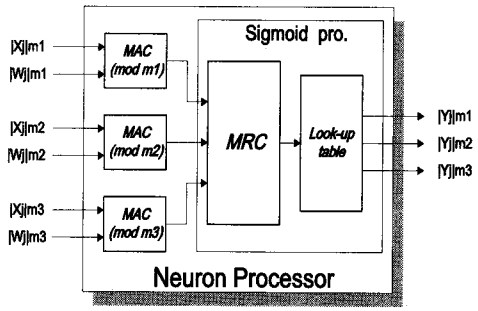
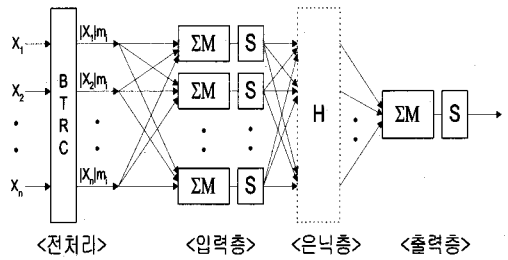


그림 17. 잉여수계를 이용한 뉴론 프로세서
Fig. 17. Digital Neuron processor using residue number system

잉여수계를 이용할 경우 모듈별로 연산이 되기 때문에, 신경회로망 시스템이 모듈리 m_1, m_2, . . . m_i, . . . m_n을 사용하는 경우 i 번째 모듈리에 해당되는 처리 과정은 그림 18과 같다.



X_1 . . . X_n 입력 정수
BTRC Binary to residue converter
$|X_k|m_i$ BTRC를 통해 잉여수로 변환된 입력
ΣM $|\Sigma(X\times W)|m_i$, multiply&sigma processing
S $|f(\ )|m_i$, sigmoid function processing
H 은닉층(hidden layer)

그림 18. i 번째 모듈리에 대한 신경회로망의 구조
Fig. 18. Architecture of neural networks for the ith moduli

그림 18은 단일 뉴론 프로세서의 구조를 나타낸다. 신경회로망 전체의 처리과정은 그림 18의 모듈별 처리과정의 반복적인 구조를 갖기 때문에 그림 19와 같이 나타낼 수 있다.

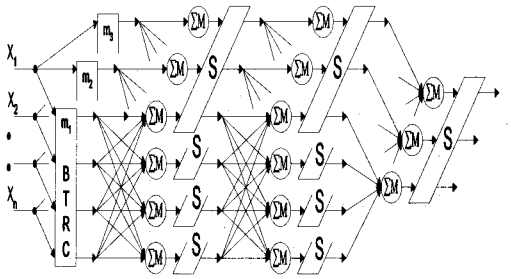


그림 19. 잉여수계를 이용한 디지털 신경회로망
Fig. 19. Digital neural network using the residue number system

V. 모의실험 및 고찰

본 논문에서 잉여수계를 이용하여 설계한 고속 디지털 뉴런프로세서는 C 언어 및 VHDL을 이용하여 논리적인 검증 및 모의실험을 수행하였으며, 최종적으로 설계된 디지털 뉴런프로세서는 Full Custom방식으로 진행되었으며, LG 0.8

μm CMOS공정을 사용하여 구현하였다. 설계한 디지털 뉴런프로세서는 일반적인 2진 연산이 아닌 11, 13, 17의 세 개의 모듈러 갖는 잉여수계의 연산을 하게 되므로 내부의 모든 데이터 흐름은 최대 5비트가 된다.

모의실험의 실행 조건으로 입력 값과 연결강도를 각각 0~10, -32~32의 정수 값으로 하였으며, 그림14의 시그모이드 함수처리를 위하여 구간 II를 9, 12, 15등분하여 실험하였다.

1. 디지털 뉴런프로세서의 설계

그림 20은 그림 9의 바렐쉬프터를 이용한 MAC연산기를 VHDL(Very High Speed Integrated Circuit Hardware Description Language)로 기술한 결과를 나타내며 그림 21은 VHDL로 기술한 MAC연산기의 입출력 파형을 나타낸다.

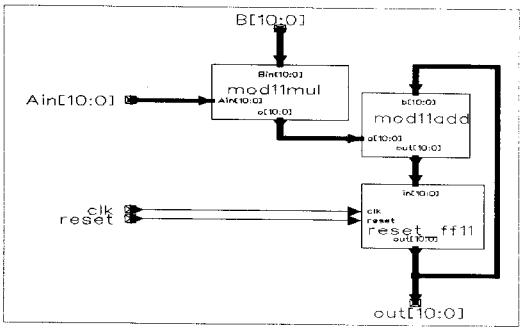


그림 20. VHDL로 합성한 MAC 연산기의 구조
Fig. 20. Structure of MAC unit synthetic using VHDL

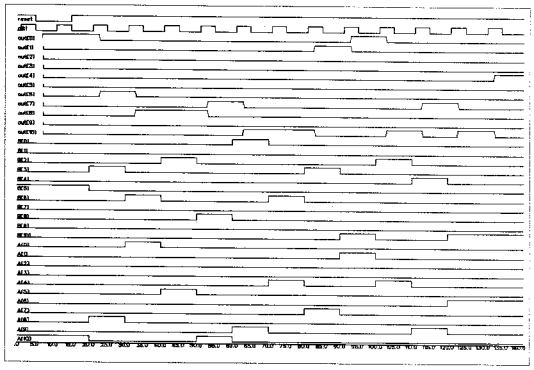


그림 21. VHDL로 합성한 MAC연산기의 입출력신호
Fig. 21. I/O signal of MAC operation unit synthetic using VHDL

그림 20은 모듈러가 11인 경우 $|out_{j+1}|_{11}=|A_{j+1}|_{11}|B_{j+1}|_{11}+|out_j|_{11}$연산을 수행하며, $|A_{j+1}|_{11}$, $|B_{j+1}|_{11}$, $|out_{j+1}|_{11}$가 각각(11^2+10^2+44)게이트로 표현되기 때문에 일반적인 정수 연산회로에 비해 작은 하드웨어로 고속의 연산이 가능하다.

그림 9의 바렐쉬프터의 연산속도를 텍스트 벡터로 작성하여 실험한 결과, 0.6ns안에 연산 결과가 출력되었다. 이 결과를 각층의 노드가 4개인 뉴런프로세서에 적용할 경우, 처음 연 산 결과를 얻는데 걸린 시간은 4.8(0.6×2×4)ns가 된다. 그림 22는 모듈러를 11, 13, 17로 할 경우 모듈러 17를 이용하여 시그모이드 함수 전체 구간을 17개로 분할하고 구간 I 를 3개(a_3=0, 1, 2), 구간 III을 3개(a_3= 14, 15, 16)로 0과 1을 출력하고, 구간 II를 11개로 정하여 시그모이드 함수 값을 출력하는 시그모이드 함수처리부의 VHDL합성결과를 나타낸다.

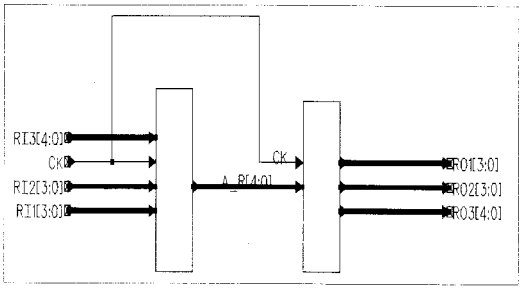


그림 22. VHDL로 합성한 시그모이드 함수 처리부의 논리회로
Fig. 22. Logic circuit of sigmoid function synthetic using VHDL

그림 23은 VHDL로 합성한 시그모이드 함수 처리부의 입출력 신호를 나타내며, 설계된 시그모이드 함수처리부는 약 16.9ns의 연산 속도와 7704여개의 Tr수를 보였다. 따라서, 바렐쉬프터를 이용하여 MAC연산기를 구성할 경우 최대 200Mhz이상으로 동작하는 연산기를 구현 할 수 있다.

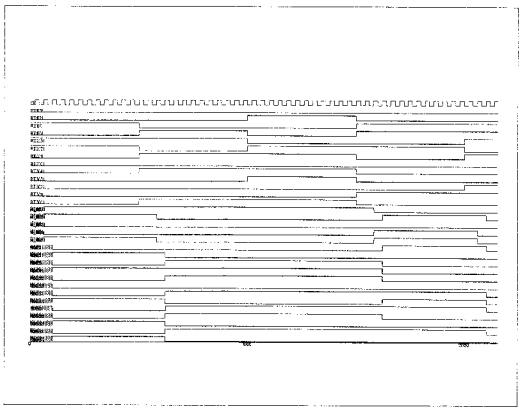


그림 23. VHDL로 합성한 시그모이드 함수 처리부의 입출력 신호
Fig. 23. Input-output signal of sigmoid function unit synthetic using VHDL

2. 디지털 뉴런프로세서의 구현

C 언어 및 VHDL에 의한 알고리즘의 검증이 끝난 후, Magic 4.1.3 Layout Tool로 레이아웃을 진행하였고 Compass V8r9으로 DRC를 수행하였다. 또한, SPICE3를 이용하여 각 연산부에 대한 논리적인 검증을 수행한 후 Irsim으로 전체동작을 실험하였다.

모의실험의 실행조건으로 표 3과 같이 순환 군이 존재하는 잉여수중, 본 논문에서 사용한 잉여 수는 모듈리 11, 13, 17로 하여 실험하였다. 설계된 PE는 약 6300개의 Tr로 구성되며, Pad를 포함하여 5m × 5m크기의 die size를 가지며, 100pin QFP Pakage로 제작되었다. 본 논문에서 구현한 뉴런프로세서의 전체 Layout는 그림 24와 같다.

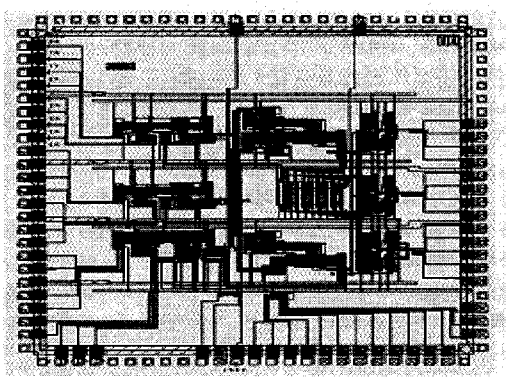


그림 24. 잉여수 체계를 이용한 뉴런프로세서의 레이아웃
Fig. 24. The layout of neuron processor using residue number system

그림 24를 이용하여 연산기를 구성할 경우, 시프트 하는데 소요되는 시간은 트랜지스터 하나를 통과하는데 걸리는 시간과 같다. 따라서, 바렐쉬프터를 이용하여 연산기를 구성할 경우, 고속연산이 가능하며, 부호정합은 표 3의 순환부호를 사용하여 입력선의 재배열로 수행되므로 논리소자가 소요되지 않는다.

그림 24에서 모듈러가 11인 경우, $|out_{j+1}|_{11}=|A_{j+1}|_{11}|B_{j+1}|_{11}+|out_j|_{11}$ 연산을 수행하며, $|A_{j+1}|_{11}$, $|B_{j+1}|_{11}$, $|out_{j+1}|_{11}$가 각각(11^2+10^2+44)게이트로 표현되기 때문에 일반적인 정수 연산회로에 비해 작은 하드웨어로 고속의 연산이 가능하다.

그림 24의 뉴런프로세서를 SPICE3로 실험한 결과, 0.6ns만에 연산 결과가 출력 되었다. 이 결과를 각층의 노드가 4개인 뉴런프로세서에 적용할 경우, 처음 연산결과를 얻는데 걸린 시간은 19.2(0.6×2×4×4)ns가 된다.

실험 결과, 실수 연산기를 이용한 뉴런프로세서와 잉여수(모듈리 11, 13, 17)를 이용하여 뉴런프로세서를 구현할 경우(수의 범위는 1024), 실수 연산기를 이용한 뉴런프로세서는 총 20652여개의 Tr로 구성되며, 잉여수(모듈리11, 13, 17)를 이용한 뉴런프로세서는 12308 여개의 Tr로 구성된다. 따라서 잉여수계를 이용한 방법이 일반적인 실수 연산기를 이용하여 구현한 뉴런프로세서에 비하여 하드웨어 크기가 50%이상 감소됨을 고찰하였다. 또한, 각 층수가 L개이고 각 층이 n개의 뉴런으로 구성되어 있는 신경회로망의 경우 Ⅲ절의 그림 4에서 설계한 1차원 어레이 의한 역전파 신경회로망의 연산시간은 다음과 같다. 기본 처리요소수가 m이고 N=n*(L-1)/m(N: 정수) 일때 1개 기본처리 요소의 연산시간이 t_{PE}이라면 N은 분할처리 수가 되며, 디지털 신경회로망의 어레이프로세스에 소요되는 연산시간 t_A는 식 (21)과 같다.

$$n=1\text{일 경우 } t_A=(n-1)*t_{PE}+m*t_{PE}$$

$$n=2\text{일 경우 } t_A=(n-1+m)*t_{PE}+((n-1-m)+m)*t_{PE}$$
$$=(n-1+m)t_{PE}+(n-1)*t_{PE}$$

$$n=k(k:\text{정수})\text{일 경우 } t_A=k*(n-1)*t_{PE}+m*t_{PE} \cdots\cdots (21)$$

본 논문에서 설계한 잉여수 및 뉴런프로세서를 이용한 어레이 구조의 역전파 신경회로망은 전체어레이 구조가 동일하므로 식 (21)과 동일한 연산시간을 갖으며 각 기본 처리요소의 연산 속도만 다르다. 즉, 기본 처리요소의 연산시간에 의한 제안된 신경회로망의 1회 연산시 연산시간은 표 5와 같다. 1개의 연산기의 연산시간을 t_c라 할 경우 MAC연산부 연산시간은 $2t_c$(잉여수 연산기 : $t_c=t_R$, 실수 연산기 $t_C=t_F$)이다. 또한, 이영수계를 이용한 신경회로망의 경우 시그모이드 함수처리부의 MRC연산은 부연산부의 층수에 비례하므로 모듈리 수를 NM이라 할 경우 $t_{MRC}=(NM-1)*t_{RNS}$의 시간이 소용되고, $t_{MRC}>2$인 경우 전체처리시간의 지연이 발생된다.

표 5. 설계된 신경회로망의 1회 연산 수행 시간
Table 5. Operation time of the designed neural network at 1 iteration

종류 \ 연산시간	t_{PE}	연산시간 (1회 수행 시)	
1차원 어레이 뉴론 프로세서	(4+j)tRNS	{k(n-1)+m+4}*tRNS	
1차원 어레이 잉여수 이용	(1+j)tRNS	$t_{MRC}\leq 2$	{k(n-1)+m+1}*tRNS
		$t_{MRC}\geq 2$	{k(n-1)+m+tMRC-1}*tRNS

표 5에서 수의범위가 작은 경우에는 잉여수를 이용한 신경회로망이 $3t_{RNS}$ 만큼 빠르고 수의 범위가 더욱 커지면 시그모이드 함수 처리부의 MRC연산의 지연($t_{MRC} \geq 5$)으로 뉴런프로세서를 이용한 신경회로망이 빠름을 알 수 있다.

VI. 결론

신경세포를 모형화한 신경회로망에 대한 이론은 현재 다양한 응용분야에서 그 우수성이 착실히 실용화되고 있다. 그러나 신경회로망이 대량의 병렬처리를 필요로 하기 때문에 하드웨어로 구현할 경우에 회로의 규모가 크고 복잡해지며 다수의 반복연산으로 인하여 속도가 저하되는 문제점을 지니고 있어서 이에 대한 대응책이 절실히 요구된다. 특히, 영상신호처리 및 패턴인식의 응용분야에서 대량의 데이터를 실시간으로 처리하여야 하는 필요성이 증가하고 있다.

본 논문에서는 신경회로망의 알고리즘 중 가장 보편적으로 이용되는 역전파 신경회로망의 연산과정에서 반복적으로 필요로 하는 행렬과 벡터의 승산과 시그모이드 함수의 연산을 고속으로 실행할 수 있는 프로세서를 설계 및 구현하였다.

첫째, 잉여수 체계를 이용한 어레이 구조로 역전파 신경회로망을 구현함으로서 기존의 실수 체계보다 고도의 병렬성을 갖는 MAC연산을 수행할 수 있었다. 또한, 잉여 수 연산은 정수연산을 수행하며, 수를 각각의 모듈리로 분리하여 연산함으로 모듈리 간에 캐리정보가 필요치 않으므로 연산기 크기가 감소하며, 고속의 연산기 설계가 가능하다.

본 논문에서 설계 및 구현한 뉴런프로세서는 1차원 어레이 구조를 갖기 때문에 입력정보의 수가 증가하여도 동일한 구조로 모듈을 파이프라인 방식으로 재 배열 함으로서 다층 구조의 역전파 신경회로망을 고속으로 실현할 수 있다.

두 번째, 순환군은 mod p-1(p:소수) 승산이 mod p의 가산과 동형이므로 부호정합에 의하여 승산기 설계가 가능하다. 그러므로 본 논문에서는 순환 부호를 이용한 잉 여수 연산 시, 소수만을 모듈리로 사용함으로써 수범 위가 확장되는 단점을 감소시켰다. 표 3의 순환부호를 사용하여 부호정합을 입출력선의 재배열만으로 수행함으로써 연산기의 크기를 감소시켰으며, 연산속도를 향상시킬 수 있었다.

세 번째, 신경회로망 구현 시 문제가 되는 시그모이드 함수 처리는 MRC를 이용하여 활성영역을 세 구간으로 분할하고 제 II구간을 다시 $2^N(N \geq 4)$으로 등분한 표본 값을 연산 표에 저장하여 두고 이용함으로서 연산표의 크기가 감소하며, 기존의 연산보다 결합강도를 큰 값으로 갱신되므로 목표치에 빠르게 수렴하였다.

결과적으로 일반적인 실수 연산기를 이용한 뉴런프로세서에 비하여 본 논문에서 설계 및 구현한 뉴런프로세서가 50%이상 연산기의 크기가 축소하며, 연산속도가 3배 이상 향상됨을 알 수 있다.

앞으로의 연구방향은 본 논문에서 구현 한 뉴런프로세서를 실제 응용화하기 위한 연구가 계속되어야 할 것이다.

참고문헌

[1] D.E Rumelhart, J. L. McClelland, etc, Parallel Distributed processing, Vol. 1., the MIT Press 1986.

[2] DARPA Neural Network study, AFCEA International press, 1987.

[3] You-Han Pao, Adaptive Pattern Recognition and Neural Networks, Addison Wesley Publishing Company Inc.1989.

[4] G. A. Carpenter, "Neural Network Models for Pattern Recognition and Associative Memory", Neural Networks ,Vol. 2, pp. 243-257, 1989.

[5] K. Fukushima, "A Neural Network for Visual Pattern Recognition", IEEE Computers, Vol. 21, no. 3, pp. 65-75, March 1988.

[6] Nicholas S. Szabo, M.S. & Richard I. Tanaka, Ph. D., Residue Arithmetic And Its Applications to computer Technology , McGRAW Hill Book Company, 1987.

[7] D. K. Banerji and J. A. Brzozowski, "On Translation Algorithm in Residue Number Systems ", IEEE Transaction on Computers, Vol. C-21, pp. 1281-1285, December 1972.

[8] L.E.Arlas and Y.Suzuki, "Digital Systems for Artificial Neural Networks", IEEE Circuits and Device Magazine, pp. 20-24, July 1990.

[9] 조원경 외 1, " 잉여수계를 이용한 디지털 신경회로의 실현", 전자공학회 논문집 제30권, B편, 2, 1993.

[10] 홍봉화외 1, "디지털 뉴런프로세서의 구현에 관한 연구", 한국컴퓨터학회 논문집 제 4권 제2호, 1999. 6.

[11] 홍봉화외 1, "시스모이드 함수의 디지털 구현에 관한

연구", 한국정보기술전략혁신학회 정보학연구 제4권 제3호 2001. 9.
[12] 서재웅 외 3, "학습된 신경망 설계를 위한 가중치의 비트-레벨 어레이 구조 표현과 최적화방법", 전자공학회 논문지 제39권 SC편 제1호, 2002. 9. pp. 37~44.
[13] 김영주 외 2, "대규모 확장이 가능한 범용신경망 연산기:ERNIE", 전자공학회 논문지 제40권 CI편 제6호, 2003. 11. pp. 56~68.
[14] 권택원 외 1, "RNS(Residue Number System) 기반의 2048비트 RSA설계", 전자공학회 논문지 제41권 SD편 제4호, 2004. 4. pp. 345~354.
[15] 홍봉화 외 1, "자기조직화 신경회로망의 학습능률 향상에 관한 연구", 한국정보기술전략혁신학회 정보학연구 제7권 제3호, 2004. 9.
[16] 홍봉화, "신경회로망을 이용한 온라인 문자인식 시스템의 자소분리에 관한 연구", 한국정보기술전략혁신학회 정보학연구 제9권 제1호, 2006. 3.

저 자 소 개

홍 봉 화(Bong Hwa, Hong)

1987년 경희대학교 전자공학과 졸업(학사)
1992년 경희대학교 대학원 전자공학과 졸업(석사)
2001년 경희대학교 대학원 전자공학과 졸업(박사)
1997년 9월 1일~2004년 2월 28일 세명대학교 컴퓨터수리정보학과 교수
2004년 3월 1일~현재 경희사이버대학교 정보통신학과 교수
관심분야 : 병렬처리, 신경회로망, 컴퓨터네트워크, etc.
E-mail : bhhong@khcu.ac.kr

주 해 종(Hae-Jong, Joo)

1988년 명지대학교 전자계산학과 졸업 (공학사)
1990년 명지대학교 대학원 전자계산학과 졸업(공학석사)
1998년 명지대학교 대학원 컴퓨터공학과 졸업(공학박사)
1997년 3월 ~ 2005년 2월 대원과학대학 멀티미디어과 조교수
2005년 9월 ~ 2006년 8월 명지대학교 IT교육센터 교수
2006년 10월 ~ 현재 인덕대학 산학협력전담교수
관심분야 : 모바일컴퓨팅, 멀티미디어품질측정, 모바일데이터베이스, etc.
E-mail : hjjoo@induk.ac.kr