論文94-31A-4-16

EWLD 알고리듬을 이용한 코드열 정합 프로세서의 설계

(The Design of a Code-String Matching Processor using an EWLD Algorithm)

趙 源 敬*, 洪 性 民**, 鞠 一 鎬*

(Won Kyung Cho, Sung Min Hong and Il Ho Kook)

要 約

본 논문에서는 코드열 정합 어레이 프로세서를 구성하기 위한 EWLD(Enhanced Weighted Levenshtein Distance) 알고리듬을 제안하고, 프로세서를 구성하는 단일 처리요소(PE)를 설계 하였다. 2차원 정합 메트릭인 EWLD 알고리듬을 1차원으로 매핑하므로서 소요되는 PE의 갯수를 N^2에서 N으로 줄일수 있다. 또한, 어레이에서 PE사이의 데이타 입출력과 PE내부의 모든 연산은 비트 시리얼로 이루어지며, 비트 시리얼 연산은 코드의 비교에 의한 코드차 WD(Word Distance)의 계산과 최단경로 선정으로서 22클럭을 1주기로 수행된다. PE의 레이 아웃은 이중메탈 1.5μm CMOS 룰에 따라 설계되었고, 1개의 PE는 약 1,800개의 트랜지스터로 구성되며 3mm X 3mm크기의 칩상에 2개의 PE를 집적 시켰다.

Abstract

In this paper we propose an EWLD(Enhanced Weighted Levenshtein Distance) algorithm to organize code-string pattern matching linear array processor based on the mappting to an one-dimensional array from a two-dimensional matching matrix, and design a processing element (PE) of the processor. N PEs are required instead of N^2 in the processor because of the mapping. Data input and output between PEs and all internal operations of each PE are performed in bit-serial fashion. The bit-serial operation consists of the computing of word distance (WD) by comparison and the selection of optimal code transformation path, and takes 22 clocks as a cycle. The layout of a PE is designed based on the double metal 1.5μm CMOS rule. About 1,800 transistors consistute a processing element and 2 PEs are integrated on a 3mm x 3mm sized chip.

* 正會員, 慶熙大學校 電子工學科
(Dept. of Elec. Eng., Kyunghee Univ.)
** 正會員, 驪州工業專門大學 電子科
(Dept. of Electronics, Yeojoo Technical College)
※ 이 논문은 1992년도 경희대학교 교비 연구비 (계열별 공모과제)의 지원으로 연구 되었음.
接受日字 : 1993年 4月 3日

I. 서론

최근 VLSI기술의 발전에 힘입어 특정 목적의 전용 프로세서 설계가 용이하여 졌으며[1]-[3], 그 결과 많은 양의 데이타를 실시간으로 처리할 수 있게 되었다. 특히 디지탈 신호처리분야에 있어서는 많은 계산량으로 인하여 실시간 처리를 위한 프로세서의 설계가 매우 중요한 것으로 인식되어지고 있다.[4]-[7]

한편, 패턴인식 분야는 인식율의 향상과 기능의 고

급화, 인식속도의 고속화를 기본으로 하여 많은 연구가 진행되고 있으며, 특히 인식율의 향상과 기능의 고급화를 위하여 처리하여야 할 데이타량은 더욱 증가하고 있는 추세이다. 여기에 처음부터 요구되고 있는 고속의 인식속도를 감안하면 현 단계에서의 가장 중요한 요소기술로서 ASIC 등에 의한 하드웨어화는 필수적이라 할 수 있다. [8]

본 논문에서는 패턴인식 분야에서 여러가지 목적으로 사용할 수 있는 코드열 정합 알고리듬을 제안하며, 이를 고속으로 처리할 수 있는 프로세서를 설계한다.

DP(Dynamic Programming) 기법을 이용한 코드열 패턴 정합 알고리듬은 이미 여러 논문[9]-[15]을 통하여 발표된 매우 잘 알려진 알고리듬이다. 코드열 정합은 임의의 두 코드열을 서로 일치시키는 과정에서 두 패턴간의 거리로서 유사도를 구할 수 있는 알고리듬이다. LD(Levenstein Distance)는 코드열 패턴의 정합에 DP(Dynamic Programming) 정합 메트릭을 적용한 것으로, 단순히 두 코드열 패턴의 최적 정합을 위한 변환의 횟수만을 패턴간의 거리로 나타낸다. [10], [12] 이에 비하여 WLD(Weighted Levenstein Distance)의 경우는 정합을 위한 변환의 방법에 차등을 두어 삭제, 삽입, 대체 변환에 의한 정합이 이루어진 경우 서로다른 가중치를 줌으로서 온-라인으로 입력되는 패턴의 최적 정합에 따른 유사도 측정에 유용하도록 한 것이다. [9], [13] 본 논문에서 제안하는 EWLD(Enhanced WLD) 알고리듬은 WLD 알고리듬을 개선한 것으로 변환 방법에 서로다른 가중치를 줄 뿐 만 아니라, 온-라인으로 입력되는 코드열임을 감안하여 서로 인접한 코드와 상대적으로 멀리 떨어져 있는 코드의 정합에도 차등의 변환 가중치를 부여한 것이다. 이와 같이 변환에 차등을 둘 경우 입력패턴의 반복적인 코드 발생에 유효하며, 인접 코드에 따라 차등을 줌으로서 온-라인 필기체 입력에서 펜의 흔들림과 같은 입력 패턴의 미세한 변화에도 대처할 수 있다.

DP를 이용한 코드열 패턴의 정합은 기본적으로 2차원 메트릭에서 정합이 이루어지므로 이를 2차원 어레이 프로세서로 설계한 경우 정합이 이루어질 코드열의 길이 N에 대하여 요구되는 PE의 갯수가 N × N으로서 매우 과도한 하드웨어가 요구된다. [10] 본 논문에서 설계한 EWLD 프로세서는 1차원으로 매핑하여 하드웨어의 요구를 N^2에서 N이하로 대폭 줄일 수 있을 뿐 아니라 상당히 긴 코드열 패턴의 분할처리에 있어서도 2차원 어레이에 비하여 상대적으로 간단히 할 수 있는 좀더 실용적인 코드열 패턴 정합용 어레이 프로세서를 구성할수 있다. 아울러 EWLD 어레이는 PE들 사이의 입출력 뿐 만 아니라 내부적으로도 모든 연산과 입출력을 비트-시리얼로 수행하도록 설계하여 칩면적당 집적도를 높이며 최소한의 회로로 시스템 구성을 용이하도록 하였다.

제안된 EWLD 알고리듬은 방향코드로 표현된 온라인-오프라인 문자패턴은 물론 물체인식등의 패턴인식에 이용할 수 있으며, 문자 또는 단어의 사전검색이나 철자검색기등에도 응용할 수 있다.

Ⅱ. EWLD 알고리듬과 1차원 어레이로의 매핑

EWLD 알고리듬은 길이가 n과 m인 두 코드열 패턴 X와 R의 정합을 수행하여 이들 두패턴 사이의 유사도를 거리값으로 계산한다. 먼저, 두 코드열 패턴 X와 R은 다음과 같이 표현된다.

입력 코드열 패턴, $X = x_1\ x_2 \cdots x_i \cdots x_n$
표준 코드열 패턴, $R = r_1\ r_2 \cdots r_j \cdots r_m$

코드열 X를 코드열 R에 일치시키기 위하여는 삭제, 삽입, 대체의 3가지 변환을 생각할 수 있으며, 다음과 같이 정의한다.

1) 삭제변환(Deletion) : $D(x_i)$

$$\alpha\ x_i\ \beta \xrightarrow{D(x_i)} \alpha\ \beta$$

2) 삽입변환(Insertion) : $I(x_i, r_j)$

$$\alpha\ x_i\ \beta \xrightarrow{I(x_i, r_j)} \alpha\ r_j x_i\ \beta$$

3) 대체변환(Substitution) : S(xi, rj)

$$\alpha\ x_i\ \beta \xrightarrow{S(x_i, r_j)} \alpha\ r_j\ \beta$$

여기서 α, β는 코드열 X와 R중의 서로 일치된 코드열이며, x_i, r_j는 일치시키고자 하는 임의의 한 코드이다. 위의 각 변환에는 변환 가중치를 각각 W_{Di}, W_{Ij}, W로 정한다. DP 알고리듬에 의한 코드열 패턴 거리값의 계산 알고리즘은 다음과 같다.

1) P(0,0) = 0;
2) FOR i=1 TO n DO P(i,o) = P(i-1,0) + W_{Di};
3) FOR j=1 TO m DO P(0, j) = P(0, j-1) + W_{Ij};
4) FOR j=1 TO m DO

FOR i=1 TO n DO

$P(i,j) = \min\{P(i-1,j)+W_D, P(i,j-1)+W_I, P(i-1,j-1)+W\}$;

5) $d(X,Y) = P(n,m)$;

LD(Levelstein Distance) [11, 12], $d^{LD}(X,R)$,은 변환 가중치를 $W_D = W_I = W_S = 1$로 고정된 값을 줌으로서 두 코드열 간의 거리는 변환의 횟수가 곧 거리값이 된다. 그러나 대체 변환의 경우 결국 연속적인 삭제와 삽입변환, 혹은 삽입과 삭제변환의 결과와 같게 된다. 따라서 세 변환에 모두 같은 고정된 가중치를 갖는 것은 DP 알고리듬을 이용한 코드열 패턴의 정합과 거리값 계산에 적절하지 않으므로 WLD (Weighted LD) [9, 13], $d^{WLD}(X,R)$,은 각 변환에 차등의 가중치를 주었다.

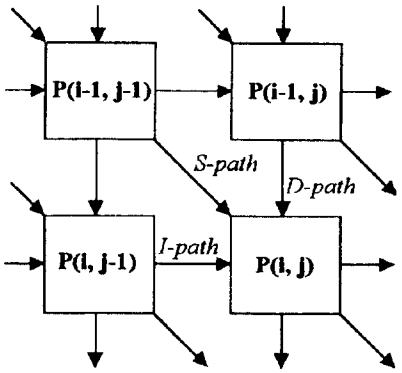


그림 1. EWLD 알고리듬의 변환경로
Fig. 1. Transformation path for EWLD algorithm.

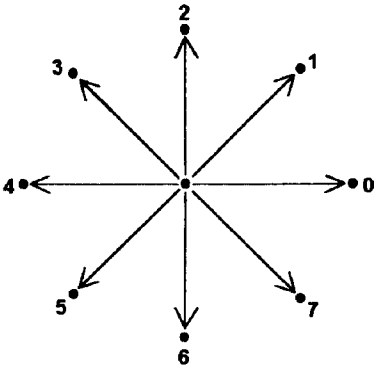


그림 2. 8 방향 코드
Fig. 2. 8-directional code.

LD와 WLD와 같은 고정된 가중치를 갖는 알고리즘은 변환이 이루어지는 코드들이 우열을 갖지 않는 코드체계일때 효과적 이다. 그러나 그림 2와 같은 8 방향 코드로서 패턴을 기술할 경우 변환의 대상이되는 코드에 우열정도 차이를 둘수 있다. 즉, 인접한 방향코드와 멀리 떨어진 방향코드와의 변환 가중치를 다르게 하므로서 8 방향 코드열로 표현한 패턴의 유사도를 좀더 세밀하게 계산해 낼수 있다. 동일한 대체변환의 경우에도 변환의 대상이 되는 방향 코드에 따라 다음과 같이 차등의 변환 가중치를 적용 하도록 한다.

$$W_{S(0,'1')} < W_{S(0,'2')} < W_{S(0,'3')} < W_{S(0,'4')},$$
$$W_{S(0,'7')} < W_{S(0,'6')} < W_{S(0,'5')} < W_{S(0,'4')},$$
$$W_{S(0,'n')} = W_{S(0,'n')},$$

예를 들어 패턴 R = 002200과 X_1 = 001100, X_2 = 006600 의 정합을 위한 변환 과정은 다음과 같다.

$$X1 \equiv 001100 \xrightarrow{S(1,2)} 002100 \xrightarrow{S(1,2)} 002200 \equiv R$$
$$X2 \equiv 006600 \xrightarrow{S(6,2)} 002600 \xrightarrow{S(6,2)} 002200 \equiv R$$

고정된 변환 가중치를 갖는 WLD에 의한 두 패턴의 거리값은 $d^{WLD}(R,X1) = d^{WLD}(R,X2)$이 되지만 가변의 가중치를 갖는 경우 $W_{S('1','2')} < WS('6','2')$ 의 변환 가중치를 적용하게 되므로 $d^{EWLD}(R,X1) < d^{EWLD}(R,X2)$로서 두 패턴의 유사도를 세밀하게 계산할 수 있게 된다.

위와 같은 가변적인 변환 가중치를 갖는 알고리듬으로는 MWLD(Modified WLD) [14], HWLD (Hamming WLD) [15]가 제안되어져 있다. HWLD 알고리듬은 각 변환에 모두 가변의 변환 가중치를 준 것으로, 그림 1은 HWLD 알고리듬의 연산을 위하여 하나의 노드에 도달할 수 있는 경로를 나타내며, 그 노드에서의 처리하여야 할 거리값 계산은 식 (1)로 정의한다.

$$P(i,j) = \min \begin{vmatrix} P(i-1,j) + WD + 1 \\ P(i,j-1) + WD + 1 \\ P(i-1,j-1) + WD \end{vmatrix}$$

이때, 세가지 변환 경로에 비교되는 코드에 따라 가변의 변환 가중치 WD(Word Distance)가 계산된다. HWLD 알고리듬의 경우 방향성분을 4-비트 존 슈코드로 표현한후 해밍 거리로서 변환 가중치를 계산한다. EWLD 알고리듬은 HWLD알고리듬의 가중치 계산 부분을 비트 시리얼 연산에 적합하도록 수정

한 것이다. EWLD 알고리듬의 경우 패턴의 표현은 8 방향성분을 3비트로 표현하여 다음과 같이 WD의 계산을 수행한다.

1) WD' = r + 2's complemnt of x;
2) if WD' > 4
then WD = 2's complement of WD'
else WD = WD';

본 논문에서 설계하는 EWLD 프로세서는 비트 시리얼 (bit serial)로 모든 처리가 이루어 지므로 위의 WD 계산 과정중 2의 보수 계산과 가산기의 구성은 1 비트 가산기와 NOT 게이트 만으로 간단히 구성할 수 있다. 따라서 단순히 3비트 2진수로 표현한 8 방향 코드의 오른쪽 쉬프트 만으로 WD를 계산과 PE 사이의 코드 전송을 수행 시킬수 있다. WD 계산부의 구성은 그림 6과 같다.

병렬성을 갖는 2차원적인 코드열 정합은 알고리듬 [10] [11] 을 그대로 실현한다면 코드열 X와 R의 각 길이의 곱 만큼의 노드 수를 갖게된다. 따라서 본 논문에서는 설계의 실효를 거두고 실현 가능하도록 하기 위하여, 노드의 수와 버스의 수를 줄이며 반복 사용이 가능한 1차원 어레이로 매핑한다.

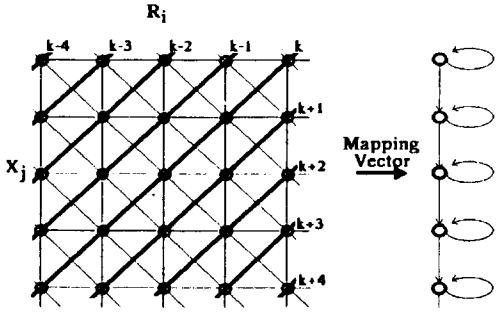


그림 3. 1차원 어레이로의 매핑
Fig. 3. Mapping into linear systolic array.

2차원으로 표현된 EWLD알고리듬을 1차원으로 매핑한 것을 그림 3에 나타냈다. 삭제변환 경로(Deletion Path)인 수직방향과 삽입변환 경로(Insertion Path)인 수평방향은 데이타의 의존성을 갖으며 사선방향은 의존관계가 없는 동시성을 나타낸다. 이들을 그림 3과 같이 매핑하면 1차원 어레이로 바꿀 수 있으며, 사선방향의 동시성은 각 노드에서 병렬처리로 실현하고, 의존성은파이프라인으로 처리한다. 다만 1차원으로 매핑할경우 k -> k+1로 처리가 진행될때 k+1의 노드들에서 필요한 I-경로와 D-경로는 k 때의 값을 필요로 함으로 파이프라인 처리에 문제가 없다. 그러나 S-경로의 경우 k+1처리를 위한 노드들은 k-1때의 계산 결과를 참조하게 되므로 k 순의 파이프라인을 위해서는 1회의 지연처리가 요구된다.

Ⅲ. 1차원 시스톨릭 EWLD 어레이 설계

EWLD알고리즘을 위한 1차원 시스톨릭 어레이는 N개의 PE로 구성되며 계산은 다음과 같이 병렬처리로 이루어 진다.

BEGIN
initial data loading
FOR k=0 to (m+n) DO IN PARALLEL for all PEs
BEGIN
Word Distance Calculation
Find Minimum Path
END
Data I/O pipeline
END

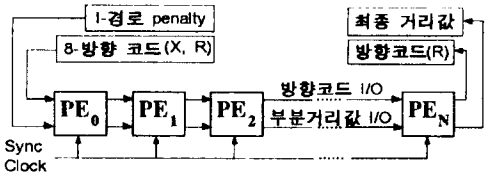


그림 4. 시스톨릭 EWLD 어레이 프로세서
Fig. 4. Systolic EWLD array Processor.

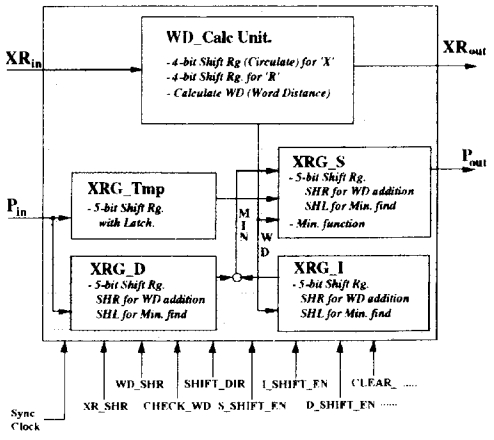


그림 5. PE의 내부 구성도
Fig. 5. Internal structure of a PE.

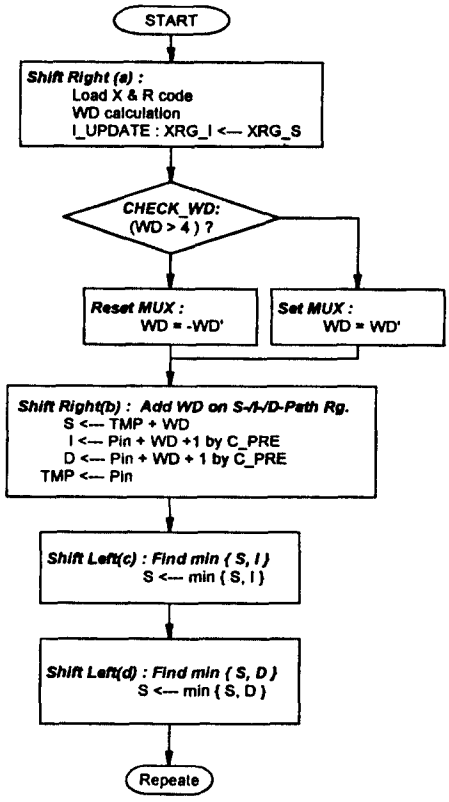
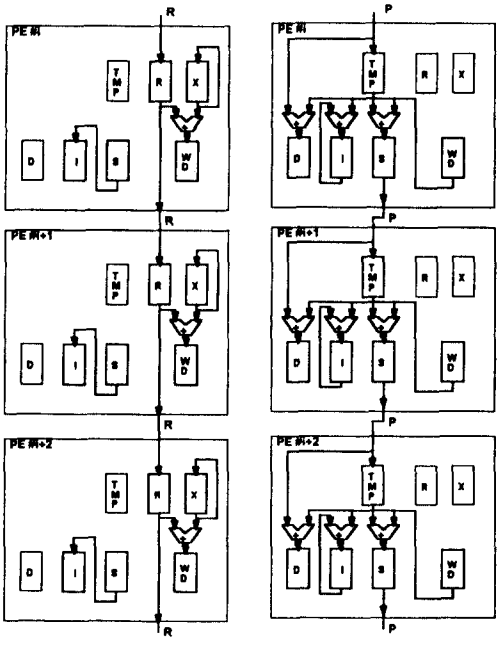
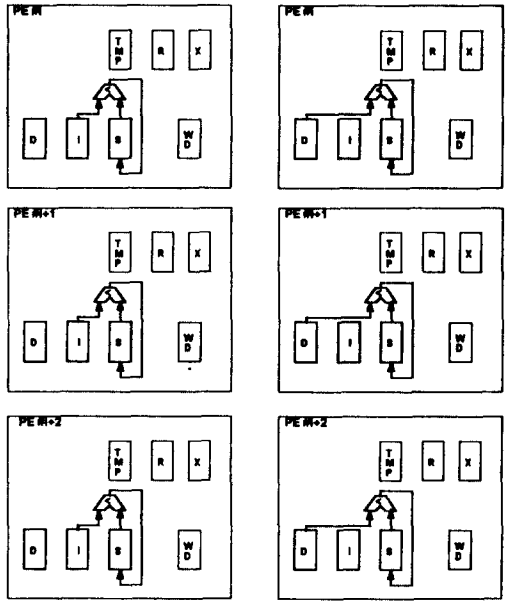


그림 6. PE의 동작 순서
Fig. 6. PE Procedure.



(a) (b)



(c) (d)

그림 7. 계산 단계별 PE내의 레지스터 전송 과정 및 PE간의 데이타 이동
Fig. 7. Register transfer of PE and Data movement between PE's.

0번째 PE에 코드열 패턴과 I-경로의 페날티 값이 공급되며 기타 PE사이에는 최적경로의 거리값과 코드열 패턴의 개별 코드가 파이프 라인 방식으로 비트 시리얼 I/O가 이루어 진다. 그림 4는 1차원 시스톨릭 어레이의 구성도이다. 1개의 PE에는 1)코드의 비교에 의한 WD의 계산, 2)최적 경로의 거리값을 찾는 부분을 포함한다. PE의 내부 구성도는 그림 5와 같다. PE의 구성 부분으로는 EWLD 알고리즘의 변환 가중치를 계산하는 WD 계산부와 세가지 변환의 변환 가중치 값을 누적 시키기 위한 세개의 경로 레지스터, 그리고 1차원으로 매핑하면서 발생하는 1회의 지연을 위한 임시 레지스터로 구성한다. 그림 6은 WD 의 계산과 최적 경로로 거리값을 찾는 PE의 동작 순서를 나타내며 그림 7은 이 동작 순서에 따른 PE 내의 처리과정을 각 단계별로 나타내었다.

1. WD 계산부

초기에 PE에 적재된 코드 R과 파이프 라인 으로 입력되는 코드 X의 거리값을 계산한다. 이때 코드의 I/O는 LSB 우선의 비트 시리얼로 이루어진다. 거리

값 WD는 8방향 코드의 상대적인 유사거리 계산에 의한 것으로 [0,4] 의 정수 값을 갖도록한다. 그림 8은 WD 계산부의 내부 구성으로서 2개의 비트 시리얼 2진 보수 계산회로를 필요로 한다.

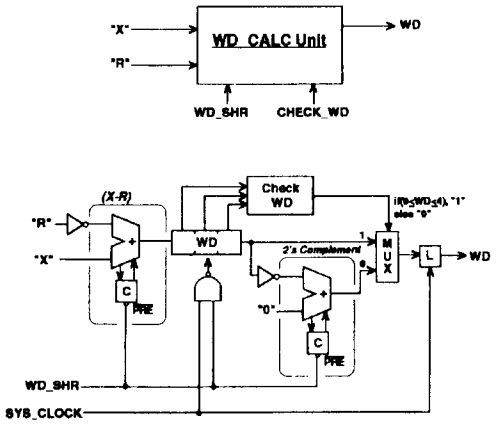


그림 8. WD 계산부
Fig. 8. WD calculation unit.

2. 최적경로 선택부

대체(S), 삽입(I), 삭제(D)등 3방향의 경로로부터 최적의 경로를 구하여 거리 값을 계산한다. 각 경로별 거리값을 위하여 3개의 레지스터와 1차원 어레이일 경우 대체 경로는 1회의 지연이 있으므로 임시 레지스터등 4개의 거리값 레지스터가 필요하다. 거리값의 I/O또한 비트 시리얼로 이루어지므로 쉬프트 레지스터이어야 한다. 특히 경로값 레지스터는 LSB우선

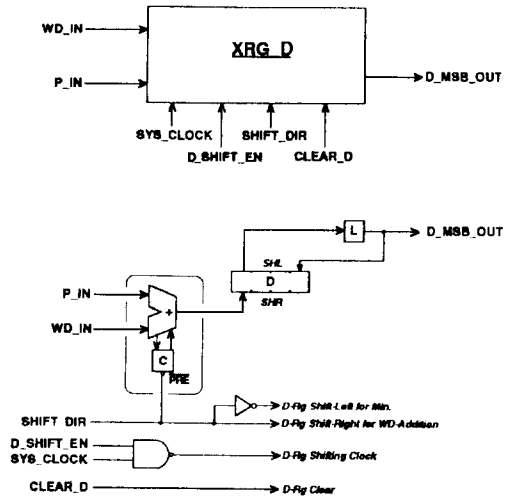
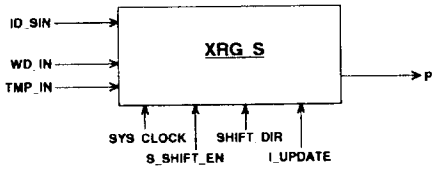


그림 9. 삭제경로 레지스터
Fig. 9. D-path Register.

비트 시리얼 I/O중에 WD의 가산이 이루어지며 최적경로를 위한 최소값을 구하기 위하여 MSB우선 되어야 하므로 경로값 레지스터는 좌우 쉬프트가 가능하도록 한다.

그림 9는 경로값 레지스터의 구조이다. 각 레지스터에는 비트 시리얼 연산을 위한 1 비트 가산기를 갖도록 함으로서 연산을 위한 별도의 제어 프로시져가 필요 없이 쉬프트 과정중에 WD의 계산과 정합 과정의 거리값 계산이 이루어 지도록 한다. 코드의 정합 과정중 누적되는 부분 거리값을 저장하게되는 레지스터의 길이를 5비트로 설계 하였다. 따라서 WD가산시 오버 플로우가 발생할 경우 레지스터의 비트 시리얼 MSB출력을 1로 고정되도록 한다.



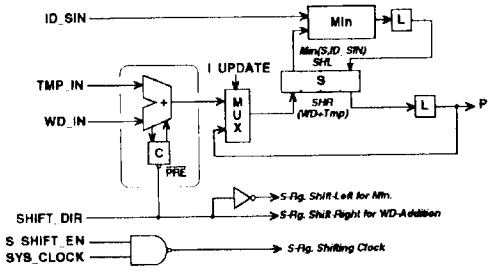


그림 10. 최소값 계산부분을 갖는 대체경로 레지스터
Fig. 10. S-path Register with min.unit.

경로 레지스터로 부터 MSB 우선 출력값을 이용하여 최소값을 찾아내면 이 값이 최적경로의 거리값이 된다. S,I,D경로중 최소값을 찾기 위하여 먼저 S와 I의 최소값을 구하여 S에 저장한후 다시 S와 D의 최소값을 구하여 S에 담는다. 따라서 최종적으로 S에 담기는 값이 최적 경로의 거리 값이다. 그림 10은 두개의 MSB 우선 입력값으로부터 최소값을 출력하는 회로를 갖는 S경로 레지스터이다.

Ⅳ. PE의 구현 및 모의 실험

1차원으로 매핑된 EWLD알고리즘은 C언어에 의한 마이크로 아키텍쳐 모의실험 기법 [16] 에 따라 PE의 동작 순서와 내부 구조를 기술하고 구현평가한

후, 이를 토대로 게이트 수준의 모의 실험을 실시하였다. 게이트 수준의 모의 실험은 HP사의 CAD 툴인 시스템 HiLo를 사용하였다.[17] 1개의 코드에 대하여 WD를 계산한 후 최적경로값을 구하기까지 소요되는 시간은 다음과 같다.

$$T_{PE} = T_{I/O} + T_{WD} + T_{min(S.D)} + T_{min(S.J)}$$

이때,

$T_{I/O}$: PE사이의 비트 시리얼 I/O소요 시간.
T_{WD} : WD의 계산 소요 시간
T_{min} : 최적 경로를 찾기 위한 최소값을 찾는데 소요되는 시간

PE의 I/O및 연산이 비트 시리얼로 이루어지므로 $T_{I/O}$, T_{WD}, T_{min}은 I/O 데이타의 비트 길이에 의하여 결정된다. 따라서,

$T_{I/O}$ = 경로값의 비트길이
T_{WD} = 코드의 비트길이
$T_{min(S.J)} + T_{min(S.D)}$ = 2*(경로값의 비트길이) + 3

가 된다. 설계한 PE의 경우 코드를 4비트로, 경로값을 5비트로 하였으며 모의 실험에서 T_{PE}는 22 클럭이 소요되는 것으로 평가되었다. 20Mhz클럭을 사용할 경우 T_{PE}는 1.1μs가 되며, 따라서 길이가 각각 m과 n인 두개의 코드열 패턴 X와 R의 거리값 계산에 소요되는 시간은 다음과 같다.

$$T_{EWLD} = T_{PE} * (m + n)$$
$$= 1.1\mu s * (m + n)$$

설계한 PE 어레이는 필기체 문자패턴의 인식 시스템 중 획 인식에 이용할 것을 목표로 하였다. 본 논문에서 사용하는 8방향 코드열로 표현된 획의 코드열 패턴의 길이는 평균 15개 정도가 되므로, 입력 패턴에 대하여 50여개의 표준패턴과의 거리값 계산에 소요되는 시간은 m은 750(=50*15), n은 15가 되어 총 841μs가 되므로 실시간 처리에 매우 유용하다.

PE의 칩 레이-아웃은 스위치 레벨의 모의실험이 가능한 레이-아웃 설계용 툴을 이용하여 1.5μ m CMOS 이중 메탈기술에 의한 디자인 룰에 따라 설계하였다. 40핀 입출력 패드를 갖는 3×3mm의 칩 면적 위에 2개의 PE를 연결하였으며 PE 1개당 약 1800개의 트랜지스터로 구성된다. 그림 11은 2개의 PE를 집적시킨 3x3mm 칩의 레이-아웃 이다. 레이-아웃은 DRC(Design Rule Check)하였으며, 동작의 검증은 2개의 PE가 집적된 레이-아웃으로 부터 회로를 추출하여 스위치 레벨의 모의 실험을 실시하였다. 특히 설계한 PE는 어레이로 구성되는 것이므로 복수개의 PE를 연결하여 동작을 검증할 필요가 있다. 이에 따라 2개의 PE가 연결된 레이아웃으로 부터 직접 회로를 추출하여 스위치 레벨 모의 실험을 실시하여 설계를 검증하였다.

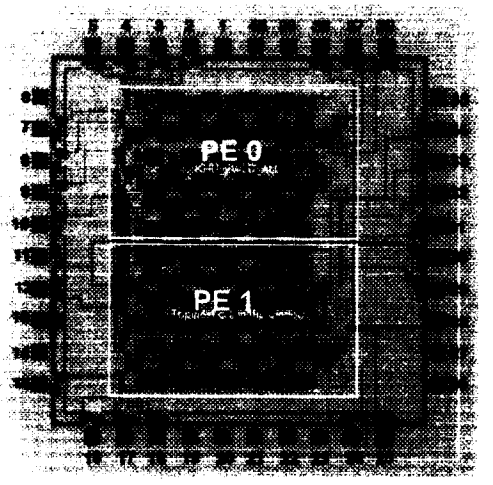


그림 11. 2개의 PE가 집적된 칩의 레이아웃
Fig. 11. Layout of PEs are integrated.

V. 결론

본 논문에서는 코드열 매칭 EWLD알고리듬을 제안하였고, 1차원 에레이 프로세서로 구성될수 있는 PE를 ASIC으로 설계하였다. DP 매칭 메트릭을 2차원 어레이 프로세서로 구성한 경우[10] 소요되는 PE의 갯수는 N^2가 요구되나 1차원 어레이로 구성하는 경우 N개의 PE가 사용 된다. 또한 설계한 EWLD 프로세서는 모든 연산과 데이타의 흐름을 비트 시리얼로 처리하므로서 최소한의 하드웨어로서 고속의 코드열 패턴 정합을 위한 어레이 프로세서를 구성할수 있도록 하였다. 실제로 EWLD 프로세서는 PC와 같은 소규모 컴퓨터에 장착되어 온라인 입력 문자 패턴의 정합에 응용할 목적으로 설계 되었으며, 현재 3mm × 3mm 칩에 2개의 PE를 집적시켜 1.5 m CMOS 기술로 제작 공정중에 있는 EWLD 프로세서는 20Mhz 클럭 속도로 동작될 경우, 1개의 PE는 초당 약 909개의 코드를 비교하여 유사도를 계산해 낼수 있다. 본 논문에서 설계한 EWLD 프로세서

는 코드열 패턴 정합에 의한 패턴 인식 분야 외에도 데이타 비트를 확장할 경우 사전검색 시스템의 전용 프로세서로서 사용될수 있다.

參 考 文 獻

[1] M.J.Foster, H.T.Kung, "The Design of Special Purpose VLSI Chips", Computer 13, pp.26-40, 1980.

[2] H.T.Kung, "Why Systolic Architectures?", *IEEE Computer*, pp.38-46, 1, 1982.

[3] Y.T.Chiang, "Parallel Processing and VLSI Architectures for Syntactic Pattern Recognition and Image Processing", TR-EE, 83-4, School of Electrical Engineering, Purdue Univ., 1983.

[4] H.D.Cheng and K.S.Fu, "VLSI Architectures for Dynamic Time-Warp Recognition of Handwritten Symbols", IEEE Trans. ASSP, Vol.34, No.3, 6, 1986.

[5] Mehdi Hatamian and Glenn L. Cash, "Parallel Bit-level Pipelined VLSI Designs for High-speed Signal Processing", *Proceeding of the IEEE* Vol.75, No.9, 9, 1987.

[6] 국일호, "코드열 패턴을 위한 시스톨릭 WLD 프로세서의 설계에 관한 연구", 경희대학교 대학원 석사학위논문, 2, 1989.

[7] M.K.Lee, B.Y.Choi, S.I.Son, "ASIC Design of a Matching Processor", KITE Journal of Electronics Engineering, Vol.3, No.1, pp.57-63, 5, 1992.

[8] 홍성민, 조원경, "문자인식 기술의 발전동향", 충남 전문대학 논문집, 제10집, pp.445-464, 9, 1992.

[9] Martin H. Ackroyd, "Isolated Word Recognition Using the Weighted Levenshtein Distance", *IEEE Trans.* ASSP, Vol.ASSP-28, No.2, pp.243-244, 4, 1980.

[10] Hsi-Ho Liu, King-Sun Fu, "VLSI Arrays for Minimum-Distance Classifications", in VLSI for Pattern Recognition and Image Processing, pp45-63, Springer-Verlag, 1984

[11] K.S.Fu, "VLSI for Pattern Recognition and Image Processing", Springer-Verlag, 1984.

[12] Eiichi Tanaka, Tamotsu Kasai, "Synchronization and Substitution Error-Correcting Codes for the Levenshtein Metric", *IEEE Trans.* Information Theory, Vol.IT-22, No.2, pp.156-176, 3, 1976.

[13] 박종진, 김은원, 조원경, "MWLD 알고리듬을 이용한 문자열정합 1차원 Bit-Serial 어레이 프로세서의 설계", 전자공학회 논문지, 제29권, B편, 제 2호, pp.1-8, 2,1992.

[14] 김현우, 홍성민, 조원경, "문자인식 시스템을 위한 코드열 정합에 관한 연구", 대한전자공학회 추계종합학술대회 논문집, 제15권, 제2호, pp.204-207, 11, 1992.

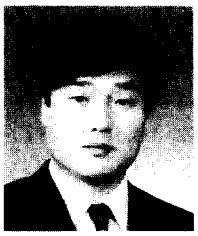
[15] 박병관, 배상덕, 서대화, 윤용호, "마이크로 아키텍춰 시뮬레이터", 전자공학회 논문지, 제24권, 제3호, pp.56-63, 5, 1987.

[16] "HP DVI for System HILO User Manual", Hewlett Packard, 1990.

著 者 紹 介

趙 愿 敬(正會員) 第 28卷 B編 第 10號 參照
현재 경희대학교 전자공학과 교수

•



洪 性 民(正會員)
1959年 9月 7日生. 1983年 2월 경희대학교 전자공학과 졸업(공학사). 1985年 8月 경희대학원 전자공학과 졸업(공학석사). 1992年 ~현재 경희 대학원 전자공학과 박사과정. 1989年 3月 ~ 1993年 2月 충남 전문대학 전자계산기과 조교수. 1993年 3月 ~ 현재 여주공업 전문대학 조교수

鞠 一 鎬(正會員)
1963年 11月 21日生. 1987年 2月 경희대 문리대 물리학과 졸업(이학사). 1989年 2月 경희대학원 전자공학과 졸업(공학석사). 1993年~현재 경희대학원 박사과정 재학중.