論文94-31B-6-2

고속 문자 인식기의 대분류용 다중 처리기의 구현

(Implementation of Multiprocessor for Classification of High Speed OCR)

金 赫 九**, 姜 仙 美**, 金 悳 鎭*

(Hyeog Gu Kim, Sun mee Kang and Duck Jin Kim)

要 約

통계적 방법을 이용하여 오프라인 문자 인식을 수행할 경우, 한글이나 한자와 같이 인식 대상 문자가 많은 경우에는 방대한 계산량으로 처리 속도가 느린 단점이 있다. 이를 개선하기 위해 인식 과정을 기능적으로 분리한 후, 각 단계를 하드웨어로 구현하여 파이프라인 처리가 되도록 구성함으로써 고속의 문자 인식기의 구현이 가능하다. 본 논문은 이러한 단계 중 대분류 과정을 고속 처리할 수 있도록 다수의 디지탈 신호처리용 프로세서(DSP)를 이용하여 병렬 구조로 구현하였다. 또한, 요구되는 처리 속도에 따라 DSP보드를 병렬로 쉽게 확장할 수 있도록 설계하였다. IBM PC에 에드 온(add-on)보드로 설계하였으며, 실험 결과 IBM PC (486DX2-66MHz)에 비해 2개의 DSP로는 약 47배, 3개의 DSP로는 약 71배의 속도 향상이 가능하였다. 구현된 다중 처리기는 본 연구실에서 구현한 고속 문자 인식기에 이용되어 그 실효성이 입증되었다.

Abstract

In case of off-line character recognition with statistical method, the character recognition speed for Korean or Chinese characters is slow since the amount of calculation is huge. To improve this problem, we seperate the recognition steps into several functional stages and implement them with hardwares for each stage so that all the stages can be processed with pipeline structure. In accordance with temporal parallel processing, a high speed character recognition system can be implemented.

In this paper, we implement a classification hardware, which is one of the several functional stages, to improve the speed by parallel structure with multiple DSPs(Digital Signal Processors). Also, it is designed to be able to expand DSP boards in parallel to make processing faster as much as we wish. We implement the hardware as an add-on board in IBM-PC, and the result of experiment is that it can process about 47-times and 71-times faster with 2 DSPs and 3 DSPs respectively than the IBM-PC(486DX2-66MHz). The effectiveness is proved by developing a high speed OCR(Optical Character Recognizer).

* 正會員, 高麗大學校 電子工學科
(Dept. of Elec. Eng., Korea Univ.)
** 正會員, 情報通信技術共同研究所
(Research Institute for Information and Communication, Korea University)
接受日字 : 1993年 10月 7日

Ⅰ. 서 론

일반적으로 통계적인 방법을 이용한 문자인식은 결정함수 이론을 바탕으로 한 것으로, 문자 영상의 위치, 방향, 밀도, 위상 정보들을 이용하여 문자 패턴을 다차원 특징벡터로 표현하고 이를 통계적 평균벡터와 비교하여 인식을 수행한다.[1] 이 방법은 초기에는 계산의 방대함, 통계적인 특성등으로 적은 종류의 패턴 인식이나 문자 인식에 이용되어 왔으나, 근래에는 VLSI 기술이 발달하여 고성능 프로세서의 개발과 ASIC을 이용한 전용 칩의 구현으로 인식 대상 문자의 수가 많고 복잡성과 유사성이 높은 한글의 인식에 있어서도 고속 처리가 가능하게 되었다.

통계적 방법에 의한 일반적인 문자인식 과정을 그림 1에 나타내었다.

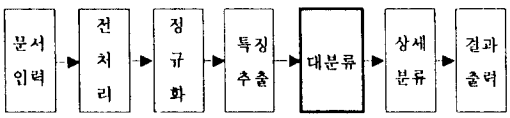


그림 1. 통계적 방법에 의한 문자인식 과정
Fig. 1. Procedure of character recognition by statistical method.

그림 1의 인식 단계별 소요 시간을 비교하기 위해 A4용지에 12포인트(point) 크기를 갖는 한글 명조체로 수록된 500자를 대상으로 문서의 입력 단계에서 인식단계 까지를 IBM PC(486DX2-66MHz)를 이용하여 C-언어로 수행시킨 결과를 표 1에 나타내었다.

표 1. 각 과정별 처리 시간의 비교
Table 1. Comparison of processing time at each of stages.

(단위: sec/500자, A4 기준)

구 분	*문서 입력	구조 분석	개별 문자 추출	정규화	특징 추출	대분류	상세 분류	합 계
수행시간 (sec)	30.0	3.5	2.3	7.3	3.3	1039.6	0.1	1086.1
비율(%)	2.76	0.32	0.21	0.67	0.30	95.73	0.01	100

*입력장치 : flat bed scanner(microtek 600z : 300dpi)

최근 발표되는 고속 문자 인식기의 특징은 파이프라인 처리나 시스톨릭 어레이 구조를 이용하여 속도를 개선한 구조가 제안되고 있다.[2,3] 본 논문에서는 인식의 고속화를 위한 단계로 표 1로 부터 전체 처리 시간의 95% 이상을 차지하는 대분류 과정을 고속화 하였다. 이를 위하여 반복적인 연산에 고속 처리가 가능한 디지탈 신호처리 전용 프로세서(DSP, TMS320C31)를 이용하여 다중처리 구조(MIMD: Multiple Instruction stream and Multiple Data stream)로 설계함으로써 처리 속도의 향상과 함께 프로그램 및 표준 데이터의 변경이 용이하며, 구조가 간단하여 확장성이 우수한 특성을 갖도록 하였다.

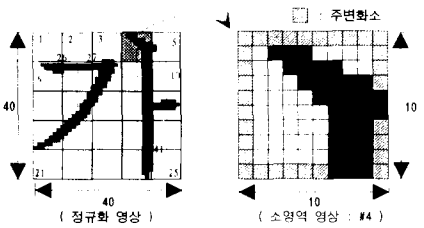
논문의 구성을 살펴보면 서론에 이어 2장에서는 대분류 알고리즘을, 3장에서는 구현된 하드웨어 및 소프트웨어의 구성에 대하여 살펴 보았으며, 4장에서는 하드웨어의 실험 결과를 제시하였고, 마지막 5장에서는 내용을 종합하여 결론을 맺었다.

Ⅱ. 대분류 알고리즘

대분류란 입력 문자에서 특징추출 과정을 통해 얻어진 특징벡터를 표준 벡터와 비교하여 유사한 후보 문자들을 선별하는 과정이다. 본 연구에서는 한글 인쇄체(2,350자)로 표준 벡터를 구성하여 인식 대상 문자의 특징벡터와 MDC(Minimum Distance Classifier)의 한 방법인 유클리드 거리 계산법을 이용하여 후보 문자를 선출하였다. 한 문자의 특징벡터 구성은 40×40 화소로 정규화된 문자 영상을 1차적으로 각 영역이 8×8화소 크기를 갖는 5×5개의 소영역으로 분할한 후, 같은 크기로 1차 영역의 일부분에 중첩되도록 4×4개로 분할하여 그림 2와 같이 총 41개의 소영역으로 구성하였다. 분할된 각 소영역에 대해 문자 영상의 윤곽선의 방향을 추출할 수 있는 템플리트를 적용시켜 4방향(→, ↓, ↙, ↘) 선소를 추출함으로써 한 문자당 총 164차원의 특징벡터를 구성하였다.

따라서, 한글 2,350자에 대한 표준 벡터의 양은 385,400개(2,350자×164차원)에 달하며, 하나의 문자를 인식하기 위하여 164차원에 대하여 2,350회의 유클리드 거리 계산을 반복 연산하도록 하였으며, 이에 대한 프로그램의 예를 그림 3에 나타내었다.

그림 3의 연산을 n개의 DSP로 병렬 처리할 경우 각 프로세서에서 계산해야 할 대상 문자의 수를 2350/n으로 줄일 수 있기 때문에 DSP간의 통신 시간을 고려하지 않는다면 n배의 속도 향상을 기할 수 있다. 이를위해 대분류 단계에서의 연산을 다수의 DSP를 이용하여 속도의 향상을 기할 수 있도록 MIMD 구조로 설계하였다.



(a)

subregion #	0° (-)	45° (/)	90° (|)	135° (\)
1	0	0	0	0
2	0	0	0	0
3	3	1	0	1
4	3	0	5	6
⋮	⋮	⋮	⋮	⋮
41	0	0	16	0

(b)

그림 2. 소영역의 분할과 특징벡터의 구성
(a) 소영역 분할(41개)
(b) 특징벡터의 예(164 차원)

Fig. 2. Seperation of the subregions and construction of a feature vector.
(a) seperation of the subregions (41 each) (b) the example of a feature vector(164 dimension).

```
for(i=0; i<2350; i++){      // 인식 대상 문자 수(KSC 5601)
  for(j=0; j<164; j++){     // 특징벡터의 차원
    dist[i] += (Std[j]-In[j])*(Std[j]-In[j]); //유클리드 연산
  }
}
for(i=0; i<2350; i++){
  계산 결과 값이 짧은 순으로 10개의 후보문자를 선택;
}
```

그림 3. 대분류 연산의 예

Fig. 3. The example of classification.

Ⅲ. 다중 처리기의 구현

1. 하드웨어의 구조

단일 프로세서에 의한 처리 속도의 한계를 극복하기 위한 방법으로 다수의 프로세서를 이용한 병렬성에 대하여 여러가지 구조가 제안 되었다. 이러한 여러가지 구조 중에서 Flynn이 제안한 명령어와 데이터에 대한 제어의 흐름에 의거하여 프로세서들이 다수의 데이터(multiple data)에 대해 다수의 명령어들(multiple instructions)을 비동기적으로 수행하는 구조를 MIMD 구조라 한다.[4,5] 이러한 MIMD 구조는 시스템의 확장성이 좋고 프로그램의 유연도(flexibility)와 적용성(applicability)이 뛰어난 장점을 가지고 있다.

본 논문은 특징벡터를 이용한 대분류 과정의 속도를 증가 시키기 위하여 공유 메모리 방식에 비해 구조가 간단하고 프로세서간의 제어가 용이하며 확장성이 좋은 비공유 메모리 구조를 이용하였으며, 상호 데이터의 교환은 공유버스 방식을 채택하였다. 그러나 공유버스 방식의 경우 프로세서의 수가 증가하게 되면 각 프로세서에 주어진 task를 수행하는 시간(R : Run time)에 비해 데이터를 전달하는데 소요되는 시간(C : Communication time)의 비가 커지기 때문에 일정 수 이상으로 프로세서를 증가 시킬때에는 오히려 성능이 저하되는 결과를 초래한다.[4,5] 구현된 시스템의 경우 대분류를 위한 실제 계산 속도는 R/N (N:프로세서의 수)배로 향상되나, 공유 버스 방식을 이용함으로써 통신에 대한 부담이 프로세서의 수에 비례(C · N)하게 된다. 따라서 전체 수행 시간은 식(1)과 같다.

$$Execution\ time = R \cdot Max(k_i) + C \cdot N \qquad (1)$$

R : task 실행 시간
C : 데이터 통신 시간 및 기타
$Max(k_i)$: 가장 오래 걸리는 프로세서의 처리시간
(균등 분배의 경우 : 1/N)
N : 프로세서의 수

프로세서 수의 증가에 따른 수행시간의 감소는 (2)와 같이 나타낼 수 있다.

$$Execution\ Time\ decrease = R\left(\frac{1}{N} - \frac{1}{N+1}\right) - C = \frac{R}{N(N+1)} - C \qquad (2)$$

$$N \equiv \sqrt{\frac{R}{C}} \qquad (3)$$

따라서 식(2)로 부터 식(3)과 같은 조건이될 때 프로세서의 수가 증가 하여도 전체 성능은 증가되지 않는 임계값을 갖는다. 따라서, R/C의 비가 크지 않는 한 프로세서의 수를 임계값 이상으로 증가시키는 것은 전체 성능에 오히려 저해 요소가 되기도 한다.

구현된 하드웨어에 사용된 프로세서는 TI(Texas Instruments)사의 TMS320C31로 33.3MFLOPS와 16.7 MIPS의 성능을 갖는 DSP전용 프로세서이다. 내부적으로 프로그램과 데이터 버스를 분리 운영

하는 Harvard architecture를 보완하여 instruction과 data fetch를 동시에 할 수 있으면서 상호 데이터 교환이 가능한 modified Harvard architecture로 구성되어 있다. 또한 일반적인 CPU에서 하나의 instruction을 수행하는데 필요한 IF (Instruction Fetch), ID(Instruction Decoder), OF(Operand Fetch), EX(Execution)의 과정들을 파이프라인 처리를 통해 하나의 cycle에 하나의 명령어가 수행 가능한 구조로 되어있다. 이 외에도 instruction cache, internal RAM, hardware multiplier, 병렬 처리 instructions, serial ports 및 DMA 기능등을 포함하고 있다. [6,7]

구현된 하드웨어는 그림 4와 같이 마스터 프로세서와 다수의 슬레이브 프로세서로 구성하였다. 마스터 프로세서는 호스트 컴퓨터와 통신을 하며, 슬레이브 프로세서로 프로그램과 표준 데이터를 전송시켜 준다. 또한, 특징벡터를 인출하여 슬레이브 프로세서로 전송한다. 전 프로세서는 입력 문자의 특징벡터와 표준 벡터를 이용하여 유클리드 연산을 수행하여 그 결과를 마스터 프로세서에서 종합한 후, 호스트 컴퓨터로 전송한다. 이에 대한 내용은 3.2절에 기술한다.

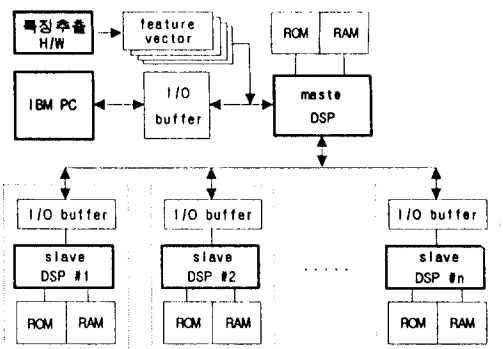


그림 4. 구현된 하드웨어의 구성도
Fig. 4. Block diagram of implemented hardware.

각 프로세서에 접속된 ROM은 DSP에서 수행할 프로그램을 호스트 컴퓨터로 부터 내부의 RAM 영역으로 다운로드하기 위한 프로그램을 저장하고 있고, RAM은 문자의 표준 데이터를 저장하기 위하여 사용하였다. 이때, 각 프로세서의 RAM에 저장되는 표준 데이터는 2장에서 언급된 것과 같이 전체 인식 대상 문자의 수를 병렬처리가 가능한 DSP의 수로 분할된 양이 저장된다. 따라서, 각 DSP에서의 계산량은 자신의 프로세서에 할당된 양에 해당되는 연산만 수행함으로써 고속화가 가능하다. 프로세서와 공유 버스간에 접속된 버퍼는 프로세서간에 통신을 하기 위한 것으로 프로그램의 다운로드, 특징벡터의 전송, 대분류된 후보문자의 전송을 위해 사용하였다.

프로세서간의 데이터 통신을 제어하기 위한 제어로직은 마스터와 슬레이브 DSP용 버퍼의 flags를 이용하여 제어하였다.

공유 버스는 8 비트의 데이터 버스와 read, write, flags 및 기타 신호(reset, clock, 전원등)들로 구성되어 있으며, 최대 6개까지 슬레이브 프로세서를 확장할 수 있도록 설계하였다.

하드웨어의 특징을 살펴보면, MIMD 구조로 대분류의 특성에 적합하도록 하드웨어를 단순화 시키면서 프로세서의 확장이 용이하도록 설계하여 시스템의 전체 성능에 요구되는 속도를 선택할 수 있도록 하였다. 또한 각 프로세서는 수행 프로그램과 입력 문자의 특징 벡터를 내부 RAM에 저장하고 표준 벡터를 SRAM에 저장함으로써 0 wait로 명령어를 수행시킬 수 있도록 하였다. 표준 벡터의 각 원소의 특징값이 1 byte를 초과하지 않기 때문에 하나의 SRAM (512Kbytes)을 사용하여 메모리의 부담을 줄였다. 데이터 통신을 위한 버퍼로는 FIFO(First In First Out)를 이용하여 고속 전송(33MByte/s)이 되도록 하였으며, FIFO의 flags를 이용하여 프로세서간의 통신을 수행함으로써 제어로직을 단순화 시켰다.

2. 소프트웨어의 구성

구현된 하드웨어에서 수행되는 처리 과정을 그림 5에 나타내었는데, 실제 시스템에서 마스터 프로세서와 슬레이브 프로세서간의 데이터 통신은 마스터 프로세서가 주관하여 순차적으로 이루어진다.

그림 5에 나타낸 대분류의 수행 과정을 살펴보면 다음과 같다.

1) 호스트 컴퓨터에서 마스터 DSP의 프로그램과 표준 데이터를 전송하면 마스터 DSP에서는 내부의 RAM 영역에는 프로그램을, 외부의 SRAM 영역에는 표준 데이터를 저장한다.
2) 마스터 DSP는 자신의 프로그램과 데이터의 다운로드가 완료되면, 병렬로 접속된 다른 슬레이브 DSP 's에도 순차적으로 호스트로 부터 프로그램과 데이터를 전송 받아 download를 수행한다.
3) 모든 DSP' s의 프로그램과 데이터의 전송이 완료되면 마스터 DSP는 특징 추출 하드웨어에서 문자 영상으로 부터 특징추출의 완료를 알리는 flag를 확인한 후 특징벡터를 읽어서, 자신을 포함한 전 슬레이브 프로세서로 순차적으로 전송한다.

4) 모든 프로세서는 특징벡터를 이용하여 표준 데이터 베이스와 식(4)에 의 거하여 유클리드 거리를 계산한다. 이때 DSP의 병렬 operations과 loop counter의 zero overhead를 이용하여 고속 연산이 가능하다. [8]

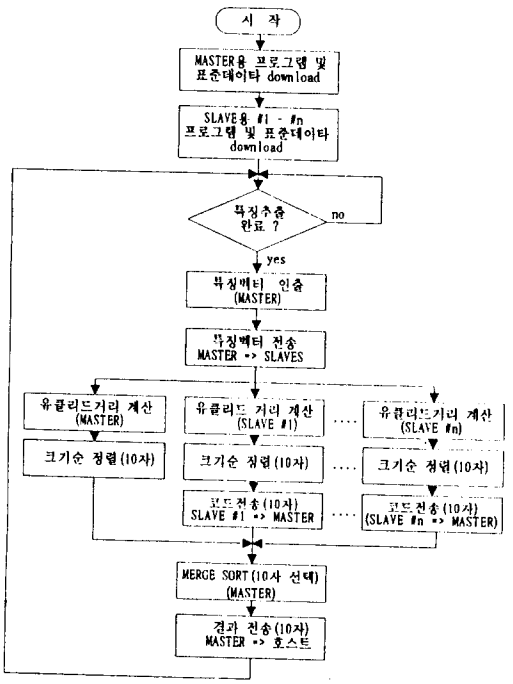


그림 5. 소프트웨어의 처리 과정
Fig. 5. Flowchart of software processing.

$$D(X,S)=\sum_{i=0}^{n}(X_i-S_i)^2 \qquad (4)$$

$D(X,S)$: $(Euclidean\ distance)^2$
X_i : 입력 문자 벡터의 i번째 원소
S_i : 표준 문자 벡터의 i번째 원소
n : 특징 벡터의 차원(164차원)

위의 식(4)를 DSP에서 처리하게 되면 뺄셈과 제곱의 연산이 한 클럭 싸이클에 수행되고, 덧셈이 다른 한 싸이클에 수행되어 모두 2클럭 싸이클에 유클리드 거리의 계산이 이루어 진다. 이를 assembly 언어로 기술하면 다음과 같다.

```
        rptb Euclidean ; zero overhead loop
                         counting
        mpy3 r1,r1,r0 ; parallel operation with
    || subi3 *ar0++,*ar1++,r1 ;
                         subtraction & multiplication
Euclidean: addi r0,r2          ; summation
```

5) 유클리드 거리 계산이 완료된 각 슬레이브 프로세서들은 거리값이 가장 작은 순으로 10개의 코드와 거리값을 정렬하여 자신의 출력 버퍼에 기록 한다.
6) 마스터 DSP는 각 스레이브 프로세서의 출력 버퍼의 flags를 확인한 후, 순차적으로 결과를 인출하여 자신의 분류 결과와 함께 병합 정렬(merge sort)을 수행한다.
7) 이로써 최종 10개의 후보 문자들의 코드와 거리값을 호스트 컴퓨터로 전송하면 하나의 문자에 대한 대분류가 완료되고, 다음 문자의 대분류를 위해 3)의 과정부터 반복하게 된다.

Ⅳ. 실험 및 고찰

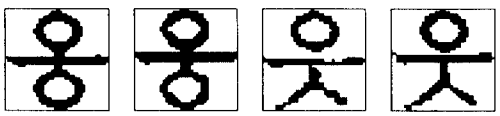
구현된 대분류 하드웨어를 실험하기 위하여, 한글 2,350자의 표준 벡터를 DSP의 SRAM에 저장시키고 입력된 문자의 특징벡터를 이용하여 유클리드 거리를 계산한 후 하나의 입력 문자에 대해 10위 까지의 후보문자를 분류하는 실험을 수행하였다. 실험에 사용된 표준 벡터는 한글 2.1 전문가용의 명조체에 대하여 5종의 크기로 레이저 프린터를 이용하여 인쇄한 후, 스캐너(300dpi)를 이용하여 읽어서 정규화를 수행한 후 특징벡터를 구성하였다. 각 크기별 특징벡터의 산술 평균값을 표준 벡터로 이용하였고, 실험에 사용된 문자 세트는 12포인트 크기를 갖는 동일 문자체를 재 인쇄와 재 입력을 통해 실험한 결과를 순위별로 누적하여 표 2에 나타내었다. 한글의 경우 그림 6과 같이 유사 문자와 문자의 인쇄 및 스캐닝 상태에 따라 문자 영상의 변형으로 대분류를 통한 후보 문자의 1위 분류율이 95.4%로 나타났으며 계산값의 일부를 표 3에 나타내었다. 그러나 유사한 후보 문자들을 선별하는 능력이 우수한 통계적 방법의 장점으로 5위까지의 누적 분류율은 99.95%로 나타났다. 따라서, 상세 분류 과정을 통해 대분류된 후보 문자들을 대상으로 입력 영상과 가장 유사한 문자를 구조적 방법으로 최종 문자를 선택함으로써 약 2% 정도 인식률이 개선되었다.

표 2. 대분류 실험 결과(한글 2350자)
Table 2. The result of Korean character classification.

대상 문자	누적 분류율(%)					
	1위	2위	3위	4위	5위	10위
명조체	95.42	99.33	99.90	99.92	99.95	100.0

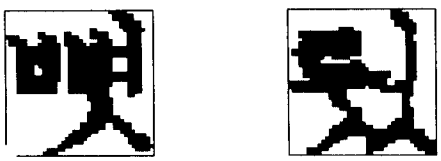
표 3. 대분류의 예
Table 3. The examples of classification.
( ) : 유클리드 거리 계산값

후보순위 / 입력문자	분류결과 1위	2위	3위	4위	5위
가	가 (273)	가 (645)	개 (795)	개 (842)	자 (877)
각	각 (238)	각 (391)	칵 (559)	칵 (686)	칵 (764)
간	간 (374)	간 (416)	칸 (676)	긴 (863)	칸 (985)
.	.	.	.	.	.
꾸	꾸 (530)	구 (735)	후 (983)	꾸 (1063)	꿈 (1125)
꾹	꾹 (567)	꾸 (621)	꾹 (803)	흑 (841)	북 (922)
꾼	꾼 (433)	끈 (570)	끈 (832)	근 (834)	근 (961)
.	.	.	.	.	.



("웅") ("웅") ("읏") ("웃")

(a)



("뻣") ("뭤")

(b)

그림 6. 오분류된 문자의 예
(a) 유사 문자 (b) 입력 영상의 결함
Fig. 6. The examples of classificatin error.
(a) similar characters.
(b) bad scanning images.

구 분	CPU 갯수	속 도 (자/초)	속도비
IBM PC	1	0.481	1
TMS320C31	1	11.375	23.7
	2	22.731	47.3
	3	34.051	70.8

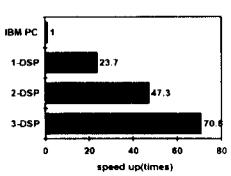


그림 7. 처리 속도 비교
Fig. 7. Comparison of processing speed.

대분류 하드웨어의 속도를 측정하기 위하여 입력 문자 500자를 A4용지에 인쇄하여 프로세서 갯수별 처리속도와 IBM PC에서 C-언어로 작성하여 처리한 결과를 그림 7에 나타내었다.

위 결과로 부터 한글 500자에 대한 계산시간 R (43.944s)과 통신시간 C(0.012s)의 비(R/C)는 약 3,662로 식(3)에 따라 병렬로 접속할 수 있는 계산상의 프로세서의 최대 갯수는 60개이다. 따라서, 통신을 고려하지 않은 계산상의 최대 성능은 1,998 MFLOPS에 999MIPS가 된다. 실험 결과 PC에서의 처리 속도보다 1개의 DSP를 이용한 경우의 속도가 약 24배로 향상되었고, 3개의 DSP's 까지 병렬 처리를 수행한 결과 거의 선형적으로 속도 향상이 이루어졌다.

V. 결 론

본 연구는 통계적인 방법에 의한 고속 문자 인식기의 구현에 있어서 가장 많은 처리 시간이 요구되는 대분류 과정의 속도를 향상시키기 위한 내용으로, 계산 속도면에서 장점을 가진 DSP(TMS320C31)를 MIMD 구조로 설계하여 병렬 처리함으로써 속도의 향상을 기하였다. 구현된 하드웨어의 구조는 비공유 메모리 방식으로 상호 접속망은 공유버스 방식을 사용하였고, 프로세서간의 통신은 버퍼(FIFO)를 사용하였다.

대분류에 소요되는 시간을 비교해본 결과 하나의 DSP 처리속도는 PC (486DX2-66MHz)의 처리 속도에 비해 24배 정도 빠르게 동작되었으며, 2개의 DSP's 에서는 약 47배, 3개의 DSP's에서는 약 71배의 속도 향상이 이루어졌다. 수행 프로그램과 표준 데이터 벡터를 호스트 컴퓨터에서 다운로드시켜 사용하기 때문에 대분류 알고리즘의 변경에 따른 프로그램과 문자체에 따른 표준 데이터의 변경이 용이하다.

구현된 대분류용 하드웨어는 병렬 처리 구조로 프로세서의 확장이 용이하기 때문에 인식 시스템의 요구되는 성능에 따라 속도 향상이 가능하여 고속 문자 인식기 뿐만 아니라, 실시간 처리가 요구되는 영상처리, 음성 처리 및 기타 여러가지 분야에 응용이 가능할 것으로 기대된다.

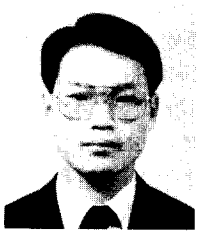
參 考 文 獻

[1] J.T.Tow, R.C.Gonzalez, *Pattern Recognition Principles*, Addison-Wesley, pp.76-78, 1974.

[2] 酒巻 久 외 5명, "認識速度 70字/秒の日本語 OCR, 專用LSIとDSPの パイプライ ン處理で高速化", *NIKKEI ELECTRONICS* (no. 505), キヤノン ソフトウェア戰略 本部, 1990.7.23.
[3] Hirotomo Aso 외 3명, "Fast and Intelligent Character Recognition System SEIUN", *일본 電子情報通信學會論文誌*, vol.J 76-D-II No.3, March 93.
[4] Harold S.Stone, *High-Performance Computer Architecture*, Addison Wesley, pp.278-312, 1987.
[5] Kai Hwang, F.A.Briggs, *Computer Architecture and Parallel Processing*, McGRAW HILL, pp.11-40, 1984.
[6] K.S.LIN, G.A.Frants, Ray Saimar, "The TMS320 Family of Digital Signal Processors", *proc. of the IEEE*, vol.75 no.9, SEP 1987.
[7] *Third-Generation TMS320* User's Guide, Texas Instruments, 1988.
[8] *Application of TMS320*, Texas Instruments, 1988.

(※ 본 연구는 삼성전자 위탁 과제로 수행 되었음)

著 者 紹 介



金 赫 九(正會員)
1961年 3月 19日生. 1984年 2月 금오공대. 전자공학과 졸업(공학사). 1990年 8月 고려대학교. 산업대학원 전자통신공학과 졸업(공학석사). 1994年 2月 고려대학교 대학원 전자공학과 졸업(공학박사). 현재 고려대학교부설 정보.통신기술공동연구소 연구원. 주관심 분야는 패턴인식, 마이크로프로세서 응용 및 영상처리 등임.

姜 仙 美(正會員)
1959年 7月 28日生. 1981年 2月 고려대학교 전자공학과. 졸업(공학사). 1988年 6月 에얼랑겐-뉘렌베르그 대학교. 전자공학과 졸업(Diplom). 1992年 8月 고려대학교 대학원. 전자공학과 졸업(공학박사). 1992年 11月- 1994年 2月 고려대학교부설 정보.통신기술공동. 연구소 연구조교수. 현재 고려대학교 산업대학원 객원조교수. 주관심 분야는 패턴인식, 영상처리 분야 등임.

金 惠 鎭(正會員) 第 29卷 A編 第 8號 參照
현재 고려대학교 전자공학과 교수