연상 메모리(CAM)를 이용한 한글 유형 분리용 칩 설계에 관한 연구

A Study on Chip Design for Hageul Type Classification using Content Addressable Memory

박 노 경*, 구 창 모**, 정 장 원*
(Noh Kyung Park*, Chang Mo Koo**, Chang Won Jeong*)

요 약

본 논문에서는 한글의 유형 분류를 CAM(Content Addressable Memory)을 이용하여 칩으로 설계하였다. 문자 인식의 전 과정을 종전의 소프트웨어에 의해서 순차적으로 처리할 경우, 실시간 처리가 가능한 고속 문자 인식기의 구현에는 어려움이 있다. 따라서, 이를 실시간으로 처리하기 위해서는 파이프라인식 하드웨어로 구현하여 시간적인 병렬성을 갖도록 하는 것이 필요하다. 하드웨어로 용이하게 구현하기 위해서 고속 병렬 매치 기능을 가진 CAM을 이용하였으며, 20개의 참조패턴만으로 유형을 분류하였다. 설계한 회로는 DAZIX의 DLAB을 사용하여 결과를 평가하였으며, 그 결과 자당 1.6μS의 처리 속도를 보였다. 또한, C-언어로 구현하여 그 결과를 비교하였다.

ABSTRACT

In this paper, we designed the chip which can classify the Korean characters using CAM(Content Addressable Memory). A high-speed OCR has been implemented by software to recognize the characters. However, it is difficult to process in real-time. The pipelined hardware implementation is one of the solution to recognize the characters in real-time by using the parallel processing techniques. We used the CAM which has the function of high-speed parallel-match to implement easily and twenty reference patterns are used for comparison. The chip has been evaluated result using DLAB of DAZIX. The simulation results have shown that the process speed was 1.6μs per character. Also, we programed using C-language and compared the results.

Ⅰ. 서 론

한글은 영문자나 숫자와는 달리 자모의 조합으로 이루어지기 때문에 그 수가 매우 방대하며, 또한 유사성을 가지고 있어 인식이 훨씬 복잡하고 어렵다. 이러한 특성을 가진 한글의 문자를 인식하기 위해서 여러가지 방법들이 시도되어 왔다. 일반적인 방법에는 원형비교방법(Templete Matching Method), 통계적인 방법(Statistical Method), 구조적인 방법(Structual Method), 신경망(Neural Method)을 이용한 방법등이 있다.[13] 원형비교방법이나 통계적인 방법은 각 문자의 특징 벡터를 미리 데이터 베이스에 구축한 뒤, 입력 문자와 비교하여 인식하는 방법으로 많은 처리 시간과 방대한 DB가 필요하다는 단점이 있으며, 신경망을 이용한 방법은 그 형태가 단순한 기능을 수행하는 처리기들이 대규모 상호 연결된 시스템을 사용하는 방법으로 필기체 문자 인식에 뛰어난 특성을 가지고 있으나 소프트웨어에 의존하여 실시간적인 하드웨어 구현의 난점이 있다. 이들 방법중에서 정형화된 입력 문자의 인식에는 하드웨어화가 용이한 구조적인 방법을 많이 사용하고 있으며, 특히 자소추출방법은 그 방법적인 면에서나 하드웨어의 구현이 매우 용이하다. 자소추출방법은 한글의 조합적 성격을 잘 반영하여, 전처리 과정에서 분리된 자모로부터 최종적으로 입력 문자를 인식할 수 있게 된다. 즉, 분리된 자모를 우선적으로 인식을 한 뒤, 그 결과를 조합하여 최종적으로 인식 결과를 출력하게 된다.

자소추출방법을 이용한 기존의 방법들은 입력문자의 유형을 분류한 뒤 세선화 및 특징점 추출을 통해서 자소를 분리하여 최종적으로 인식하는 방법들을 취하였다. 이중에서 고속문자인식기의 구현을 위하여 세선화 및 특징점 추출은 하드웨어화를 실현하였으나, 유형분류는 소프트웨어에 의한 호스트 컴퓨터에 의존하는 형태였다. 이 경우 병렬처리컴퓨터를 사용한다 할지라도 처리속도

*호서대학교 정보통신공학부
**고려대학교 전자공학과
접수일자 : 1995년 11월 27일

에는 한계가 있으며, 하드웨어화에 의한 실시간 처리와의 병목현상까지도 유발할 수 있다. 따라서 본 논문에서는 이러한 난점을 하드웨어화로 극복하고자 CAM을 이용하여 유형 분류용 칩을 설계하였다.

분류 방법은 입력된 문자를 유형에 따른 고유패턴의 조건에 따라 분류할 수 있는 판별트리를 사용하였다. 판별트리를 이용하여 칩으로 설계를 하기 위해서는 입력문자와 고유패턴과의 상호 비교가 필요하게 되는데, 기존의 메모리를 이용할 경우 어드레싱으로 인한 속도의 저하를 야기시킨다. 본 논문에서는 이러한 점을 보완하기 위해서 메모리 자체내에 고속병렬 매치(match)기능을 가진 CAM (Content Addressable Memory)[1, 2, 3, 4, 5]을 이용하였다. 또한 칩으로 설계를 함으로써 병렬 처리가 가능해지는 이점이 있다. 설계한 회로는 모의실험을 통해 결과를 평가하였으며, C-언어로 구현하여 그 결과를 비교하였다.

본 논문의 구성은 2장에서 한글의 유형 분류, 3장에서는 판별 트리 구조, 4장에서는 회로로 설계한 방법을 나타내었으며 5장에서는 설계한 회로의 실험 결과 및 검토, 6장에서는 결론을 기술하였다.

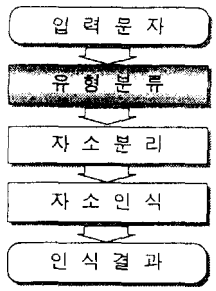


그림 1. 자소 분리를 이용한 문자 인식 흐름도
Fig 1. Flowchart of character recognition using Jaso segmentation

Ⅱ. 한글의 유형 분류

한글 문자의 형태 분류는 문자를 크게 그룹지어 그 그룹의 소속을 판별하는 것이므로 인식을 위한 전 단계라고 볼 수 있다. 한글 문자는 구조적 특성인 모음의 형태와 받침의 존재 유무에 따라 일반적으로 6가지 형태로 조합하여 표현할 수 있으며 인쇄체 한글의 경우 각각의 조합 형태에 대한 초성, 중성, 종성의 위치가 거의 일정한 위치에 놓이게 된다.

한글 문자는 수평선(−), 수직선(|), 사선(/) 등의 직선적인 성분과 원(○) 및 굵기에 의하여 구성되어지며 이러한 구조 및 위치의 특징에 의하여 그 유형을 판별할 수 있다. 그림 2는 한글의 구조적인 특징을 사용하여 6가지 유형으로 분류한 것이다.[7, 8, 9]

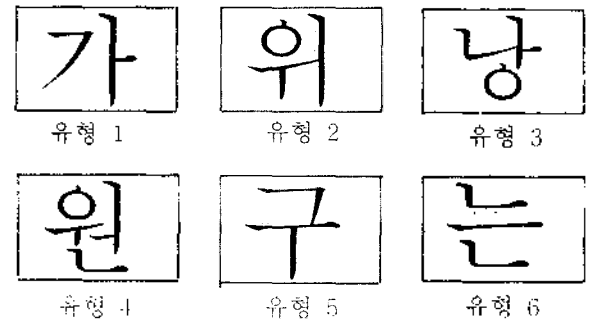
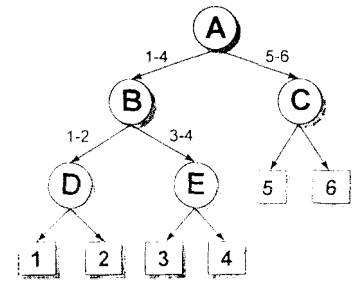


그림 2. 한글의 유형
Fig 2. Type of Hangul

Ⅲ. 판별 트리 구조

판별 트리 구조는 한글 문자의 유형 분류를 위한 결정적인 역할을 담당하는 것으로서, 한글 고유의 모음 위치와 자음 위치(그림 2. 참조)를 판별하여 그 유형을 결정짓는다. 판별 트리는 계층적으로 구성된 2진 트리로 구성되어 있으며, 5개의 노드를 가지고 있다. 각 노드에는 그 특징에 따라 일정한 유형을 판별하게 된다. 이 일정한 유형을 가지고 참조 패턴(reference pattern)을 생성하여 입력된 문자와의 비교를 위해 고속 병렬 매치 기능을 가진 CAM을 이용하였다. CAM은 일종의 메모리로서 기존의 메모리 기능을 가지고 있는 동시에 저장되어 있는 데이타를 어드레싱없이 메모리 자체에서 고속으로 병렬 매치시킬 수 있다. 이러한 CAM의 특징을 이용하여 참조 패턴과 저장되어 데이타를 매칭(matching)시켜 유형을 판별하게 된다.



A 노드 : 수평 모음(ㅗ, ㅛ, ㅜ, ㅠ, ㅡ)의 유무로 1-4형과 5-6형을 판별
B 노드 : 수직 모음(ㅏ, ㅑ, ㅓ, ㅕ, ㅣ)의 유 무로 1-2형과 3-4형을 판별
C 노드 : 수평 자음을 가진 것은 같으나 받침 자음의 유무로 판별
D 노드 : 이중 모음(ㅚ, ㅟ, ㅘ, ㅙ, ㅞ, ㅝ, ㅢ)의 유무로 1형식과 2형식을 판별
E 노드 : 받침이 있는 형식은 같으나 이것 역시 이중 모음으로 판별

그림 3. 유형 결정을 위한 판별 트리 구조
Fig 3. Tree structure for decide on type of Korean language

판별 트리에서 사용되는 참조 패턴은 모음들의 모양과 발생되는 위치로 생성하였다. 여기서 필터는 유형에 따라 발생되는 모음의 위치와 길이를 나타낸 것으로서 보다 정확한 결과를 위해 사용하였다. 그 각각의 노드들이 가지고 있는 참조 패턴을 살펴보면 다음과 같다. 먼저 노드 A에서는 다른 형식에서 찾아볼 수 없는 긴 수평 모음을 가지고 있으며, 그 발생 위치와 형태는 그림 4와 같이 유사하다. 노드 B에서는 수직으로 긴 모음을 사용하였으며, 참조 패턴과 필터의 위치는 그림 5와 같다.

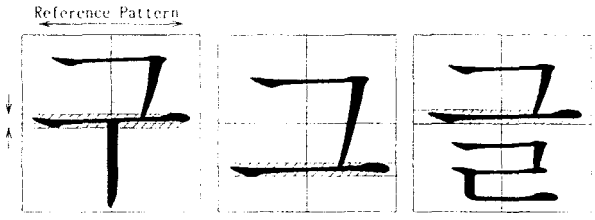


그림 4. 노드 A에 대한 참조 패턴과 필터 위치
Fig 4. Reference patterns and filter positions for node A

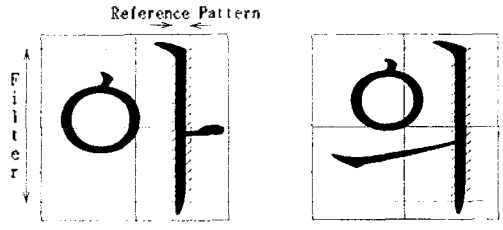


그림 5. 노드 B에 대한 참조 패턴과 필터 위치
Fig 5. Reference patterns and filter positions for node B

노드 C에서는 5 형식에서만 사용하는 모음(ㅗ, ㅛ, ㅜ, ㅠ, ㅡ)를 이용하였다. 이 경우 6 형식에서 사용하는 것과 비교할 경우 수직으로 뻗은 길이가 월등히 긴 것을 알 수 있다. 또한 중성 모음 'ㅡ'는 5 형식에서만 나타나는 특징이므로 이를 사용하여 참조 패턴을 추출하였다. 그림 6은 노드 C에 대한 참조 패턴과 마스크의 위치를 나타낸 것이다.

노드 D에서는 2 형식에서만 나타나는 이중 모음(ㅚ, ㅟ, ㅘ, ㅙ, ㅝ, ㅞ, ...)를 이용하였으며, 그림 7에 나타내었다. 여기서 '의, 봐, 왜' 등과 같은 문자는 이중 모음의 발생 위치가 유사하여 필터의 수를 줄일 수 있다. 마지막으로 노드 E에서는 노드 D에서 사용한 것과 유사한 방법으로 참조 패턴을 추출하였다. 즉, 4 형식에선 나타나는 이중 모음을 이용하였다. 여기서 노드 D와 다른 것은 마스크의 위치이다. 노드 E에서 이중 모음의 위치는 노드 D에서 발생하는 것보다 높은 곳에서 발생한다. 그림 8은 노드 E에 대한 참조 패턴과 필터의 위치를 나타내고 있다.

본 논문에서는 설계한 칩의 크기(size)와 최소한의 동작

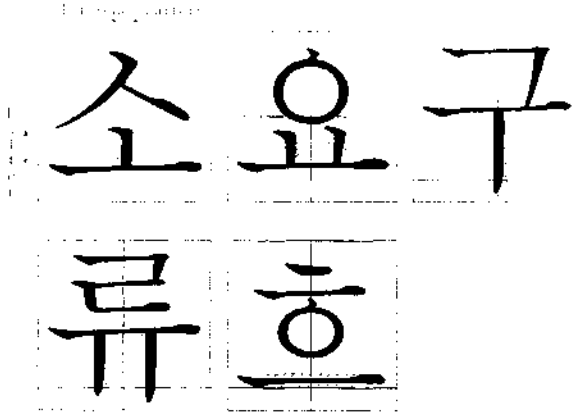


그림 6. 노드 C에 대한 참조 패턴과 필터 위치
Fig 6. Reference patterns and filter positions for node C

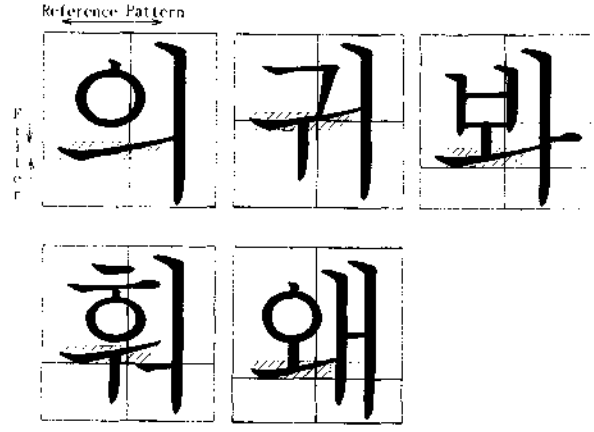


그림 7. 노드 D에 대한 참조 패턴과 필터 위치
Fig 7. Reference patterns and filter positions for node D

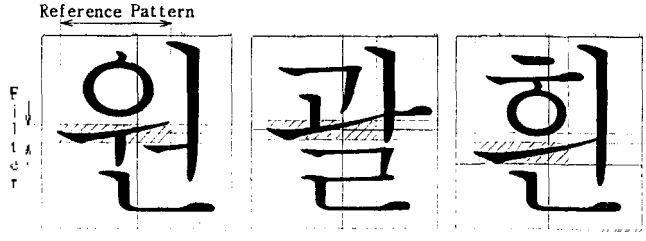


그림 8. 노드 E에 대한 참조 패턴과 필터 위치
Fig 8. Reference patterns and filter positions for node E

속도를 고려하여 20개의 참조 패턴을 사용하였으며, 표 1에 참조 패턴과 필터의 위치를 나타내었다. 표 1에서 일련 번호는 비교 순서를 나타낸다. 인쇄체 문자를 대상으로 생성한 참조 패턴과 필터를 가지고 입력된 문자와의 비교를 통해 발생한 결과는 참조 테이블에 비교 순서대로 저장된다. 그림 9에서는 회로를 설계하기 위한 전체적인 흐름도를 나타내었다.

표 1. 참조 패턴 및 필터의 위치
Table 1. Reference Pattern & Filter position

순 서	참조패턴(열)	필터의 위치(행)
1	5 - 35	16 - 20 / 29 - 32
2	27 - 28	4 - 33
3	28 - 29	4 - 33
4	30 - 31	4 - 33
5	17 - 18	24 - 32
6	19 - 20	24 - 32
7	20 - 21	24 - 32
8	14 - 15 / 24 - 25	21 - 33
9	5 - 35	30 - 32
10	11 - 12	22 - 27
11	12 - 13	22 - 27
12	13 - 14	22 - 27
13	14 - 15	22 - 27
14	15 - 16	22 - 27
15	3 - 15	27 - 30
16	13 - 14	24 - 32
17	16 - 17	24 - 32
18	17 - 18	24 - 32
19	15 - 16	24 - 32
20	5 - 11	19 - 24

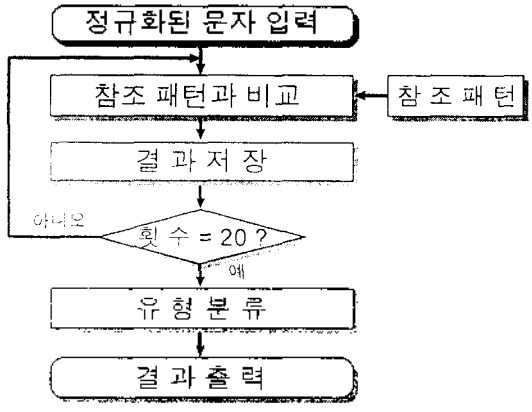


그림 9. 유형 분류의 흐름도
Fig 9. Flow chart of Hangul type classification

Ⅳ. 설 계

한글의 유형 분류를 소프트웨어가 아닌 하드웨어로 설계하기 위해서 본 논문에서는 고속의 병렬 매치 기능을 가진 CAM을 이용하였다. 문자 인식기에서 입력된 영상(문자)을 처리하기 위해서는 우선적으로 메모리에 저장을 해야만 한다. 따라서 한글의 유형 분류를 위한 하드웨어를 별도로 만들 경우 그 만큼의 오버헤드가 발생하게 된다. 그러나 입력 데이터를 저장한 메모리를 그대로 이용한다면 훨씬 더 효율적일 것이다. 또한 참조패턴을 이용하기 위해서는 입력된 데이터와 참조패턴과의 비교가 수행되어야 한다. 본 논문에서는 이러한 점을 가능케 하기 위해서 읽기, 쓰기 및 매치 기능을 가진 CAM을 사용하였다. CAM은 입력 데이터의 읽기, 쓰기와 함께 외부에 별도의 비교기가 없이 자체적으로 저장된 데이터와 입력된 데이터를 비교할 수 있다. 따라서 참조 패턴을 사용하여 유형을 분류하는데 적합하다.

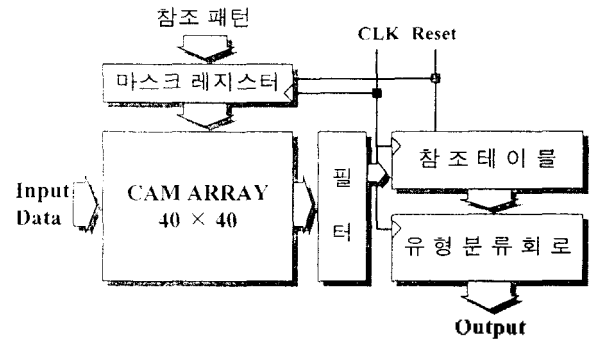


그림 10. 전체 시스템 블럭도
Fig 10. System Block Diagram

CAM을 사용한 궁극적인 목적은 바로 참조패턴을 보다 효율적으로 사용하기 위함이다. 즉, 참조패턴과 입력 데이터를 비교해야하는 과정은 필연적이므로, 비교 기능이 탁월한 CAM을 이용하여 비교 기능을 수행하기 위한 과정을 줄일 수 있게 하였다.

설계에 사용된 툴(tool)은 DAZIX의 Aceplus로 Xilinx 라이브러리를 사용하여 schematic으로 그렸다. 시스템은 40×40 크기의 정규화된 입력 문자를 고려하여 40word×40bit의 CAM과 레지스터부 그리고 조합 회로로 구성되어 있다. 그림 10에 나타낸 전체 시스템의 동작 과정을 설명하면 다음과 같다. 먼저 Reset 신호에 의해 필요한 부분들을 클리어(clear)한 후, 유형 분류가 수행될 데이타를 외부로 부터 40비트씩 병렬로 읽어들여 CAM에 저장한다. 저장이 끝나면 CAM에 저장된 데이타를 참조 패턴과 비교하여 매치(Match) 여부를 감지하게 된다. 매치 여부는 필터를 통하여 참조 테이블에 저장한 뒤, 최종적으로 유형 분류회로에서 참조 테이블을 참조하여 입력된 데이타의 유형을 출력하게 된다. 다음은 각 블럭의 기능을 설명하고 있다.

4.1 CAM

CAM은 SRAM과 유사하여 읽기(read)와 쓰기(write) 기능을 모두 가지고 있으며, 여기에 매치 기능이 첨가되어 있는 메모리이다. 매치 기능은 입력 워드와 저장된 워드를 비교해서 매치시키기 위해 병렬로 검색하며, 매치가 됐는지 않됐는지를 지적한다.

그림 11은 게이트 레벨로 표현한 CAM 셀이다. 여기서 CAM 셀의 동작을 살펴보면, 우선 쓰기 동작은 Wi 신호

가 high가 될 때 Bi의 입력 데이타가 비트 저장(bit-storage)을 구성하고 있는 G3와 G4 게이트에 저장된다. 저장을 끝내려면 Wi 신호를 low로 하면 된다. 매치 동작은 Wi 신호가 low로 set된 상태에서 G3와 G4에 의해 저장된 데이타와 Mi의 입력을 비교하는 것으로서, 만약 저장된 데이타와 Mi 입력이 같으면 출력 Mo가 high가 되고, 그렇지 않으면 low가 된다. 이와 같이 설계된 회로에서는 읽기 동작이 필요하지 않으므로 읽기 동작을 위한 게이트를 제거하였다. 그 결과 게이트의 수를 7개에서 6개로 줄일 수 있었다.

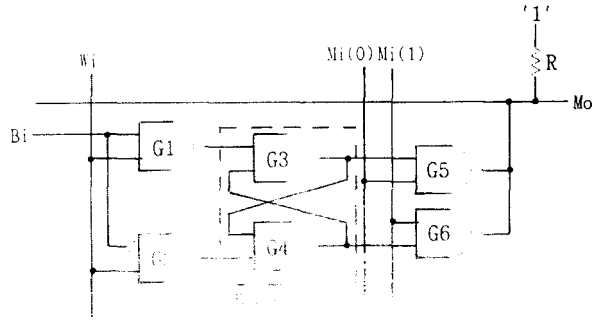


그림 11. CAM 셀 논리 회로도
Fig 11. Logic citcuit of CAM cell

4.2 마스크 레지스터

마스크 레지스터는 매치 동작시에 외부(주 메모리)로부터 참조 패턴을 받아서, CAM으로 출력시켜주는 역할을 한다. 즉, 클럭과의 동기를 위해 CAM의 매치 입력 전단에 삽입하였다. 마스크 레지스터의 크기는 워드(word) 수를 고려하여 40 비트의 PIPO 레지스터를 사용하였다.

4.3 참조 테이블

참조 테이블은 CAM에서 매치된 결과를 저장할 때 필요한 회로로서, 필터를 통해 나온 결과를 순차적으로 저장한다. 따라서, 20개의 참조 패턴을 사용하였으므로 20비트의 SIPO 레지스터로 구성하였다. 참조 테이블의 저장된 내용은 유형을 결정하는데 참조된다. 그림 12는 참조 테이블에 할당된 각 노드들의 위치를 나타내고 있다.

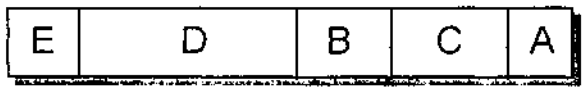


그림 12. 참조 테이블
Fig 12. Reference Table

4.4 필 터

필터는 CAM의 특정한 영역(일정한 행)에서 매치가 발생한 것만을 감지하도록하여 보다 더 정확한 결과를 가져오도록 하였다. 즉, 각 노드에서 매치가 발생되는 위치(표 1)만을 감지하며, 그 외의 장소에서 매치가 발생하면 무시하므로서 보다 더 정확한 결과를 가져올 수 있게 하였다. 이 회로는 입력이 모두 '1'일 때 출력이 '1'인 기능을 가지는 AND 게이트를 사용하였다.

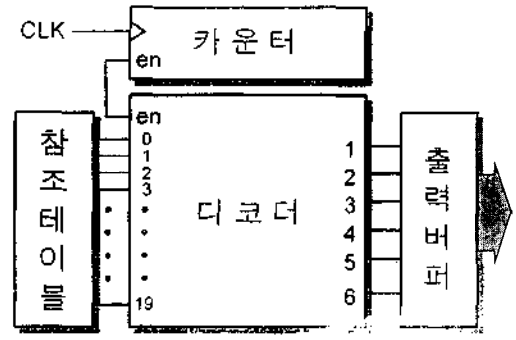


그림 13. 유형 분류 회로의 블럭
Fig 13. Block diagram of circuit for type classification

4.5 유형 분류 회로

유형 분류 회로의 역할은 일종의 디코더(decoder)와 카운터(counter)를 조합한 회로이다. 동작을 살펴보면 먼저 카운터(counter)는 클럭의 수를 계산하여 20번의 수행을 마치면 en 신호선이 high가 되며, 이 동작 신호(en)에 의해 디코더는 참조 테이블에 저장된 내용을 참조하여 최종적인 입력 데이터(문자)의 유형을 출력하게 된다. 이때 해당되는 출력핀만이 high가 되며 나머지는 low를 유지한다. 그림 13는 유형 분류 회로를 블럭으로 나타낸것이다. 여기서 디코더의 입력은 참조 테이블(20bit)과 연결된다. 표 2는 디코더가 입력 문자의 유형을 판별하기 위해 사용되는 분류표이다.

표 2. 참조 테이블을 참조한 유형 분류표
Table 2. Table of Hangul type classification refered to reference table

노드 / 형식	A	B	C	D	E
1 형식	0	1	X	0	X
2 형식	0	1	X	1	X
3 형식	0	0	X	X	1
4 형식	0	0	X	X	1
5 형식	1	0	1	X	X
6 형식	1	0	0	X	X

'1' : Match '0' : Mismatch 'X' : don't care

V. 실험 결과 및 검토

본 연구에서는 정규화 과정을 거친 40×40 크기의 문자를 대상으로 하였으며, 문서에서 많이 사용되는 완성

형 명조체와 윈도우즈용 바탕체를 사용하였다. 실험에 쓰인 대상 문자는 각 유형에 대해서 완성형으로 표현할 수 있는 무작위 표본 500자를 대상으로 하였다. 유형 분류 인식율을 실험하기 위해서 C-언어로 구현하였으며, 그 결과 분류율은 98.5%로 나타났다. 오분류된 원인을 고찰해 보면, 주로 2 형식과 6 형식 사이에서 나타났으며, 그 원인이 되는 글자에는 모음 'ㅛ'가 포함되어 있었다. 즉, 1 형식인 '과'가 5 형식으로 2 형식인 '팔, 판, 펄'이 6 형식으로 각각 오분류 되었다. 이것은 'ㅛ'의 하변이 이중 모음(ㅢ, ㅘ, ㅝ, ...)과 거의 유사한 길이를 가지며, 그 발생 위치 또한 유사한데서 오는 결과로 추정할 수 있다. 아래의 프로그램은 C-언어로 구현한 main 부분을 나타낸 것이며, 그림 14에 C-언어로 구현한 것을 이용하여 유형을 분류한 예를 보이고 있다. 여기서 유형 분류의 결과는

```
void main(void)
{
  int MATCH(int, int, int);
  void TEST_PATTERN(void);
  void FILTER(void);
  void SCAN_TABLE(void);
  void PRINT_OUTPUT(void);
  init();
  TEST_PATTERN();
  file_read();
  for (i = 0; i <= 39; i++)
        for (j = 0; j <= 39; j++)
                MATCH(REFERENCE_BIT[i][j], REFERENCE[i][j], INPUT[i][j]);
  }
  FILTER();
  SCAN_TABLE();
  PRINT_OUTPUT();
}
```

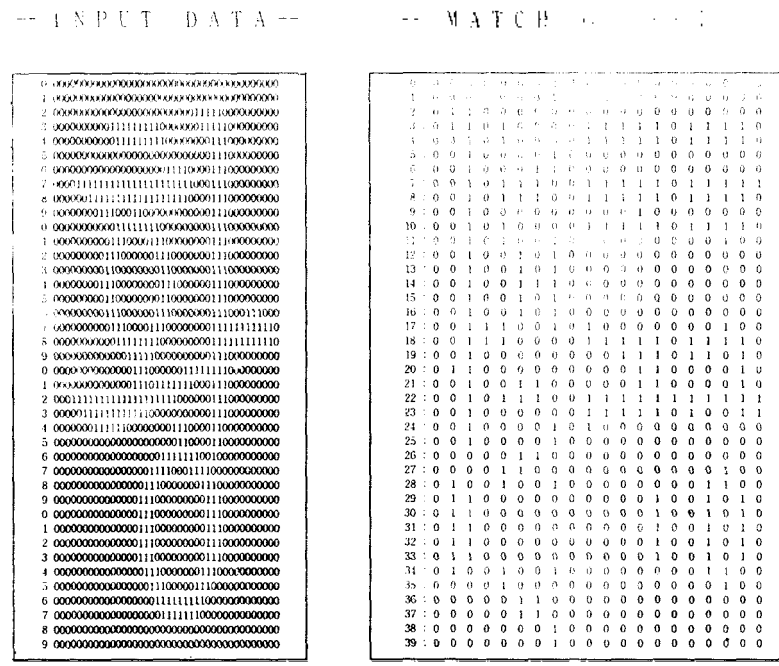


그림 14. 유형 분류의 예
Fig 14. Example of Hangul type classification

참조 테이블을 참조하여 얻은 것이다.

유형을 분류하는데 소요되는 시간을 IBM PC(486DX2-66MHz)와 팬티엄(90MHz) 및 워크스테이션(SPARC 2)에서 C-언어로 구현한 소프트웨어와 비교하여 표 3에 나타내었다. 이 결과를 보아 소프트웨어보다 월등히 빠른 수행 속도를 나타내고 있음을 알 수 있다. 표 4는 각 블럭별 회로에 대해 SPICE를 이용한 시뮬레이션 결과를 보이고 있다. 이 표에서 알 수 있듯이 가장 큰 비중을 차지하는 것은 CAM 블록이며 11.2ns의 빠른 매치 시간을 나타내고 있다. 총 지연시간에 따른 최대 동작 주파수는 약 27MHz이다.

표 3. 유형 분류 소요 시간 비교
Table 3. Comparison of type classification time

실험에 사용된 시스템	소요 시간
IBM PC(486DX2-66MHz)	86.7 ms/자
팬티엄(90MHz)	65 ms/자
워크스테이션(SPARC 2)	58.6 ms/자
설계한 CAM 시스템	1.6 μs/자

표 4. 각 블럭별 회로 시뮬레이션 결과
Table 4. Circuit simulation result for each blocks

회 로	지연시간	총 지연 시간
마스크 레지스터	5ns	5ns
CAM(저장)	18.8ns	---
(매치)	11.2ns	16.2ns
필터	7.7ns	23.9ns
참조 테이블	6ns	29.9ns
유형 분류 회로	6.6ns	36.5ns

시뮬레이션에 사용된 주파수는 25MHz를 사용하였으며, 그 결과 데이타를 읽는 시간이 약 800ns정도 소요되었으며, 실제 유형을 분류하는데 소요되는 시간은 약 840ns가 걸렸다. 따라서 전체 소요 시간은 약 1.6μs 정도이며, 기존에 제시된 세선화 및 특징점 추출을 통한 과정을 경우할 경우 약 80μs 정도의 처리 시간[9]보다 월등히 빠른 처리 속도를 나타내고 있다. 그림 15는 설계된 회로의 임계 경로(critical path)를 사용하여 실험한 결과를 나타낸 것이다. 즉, 하나의 참조 패턴이 CAM으로 입력된

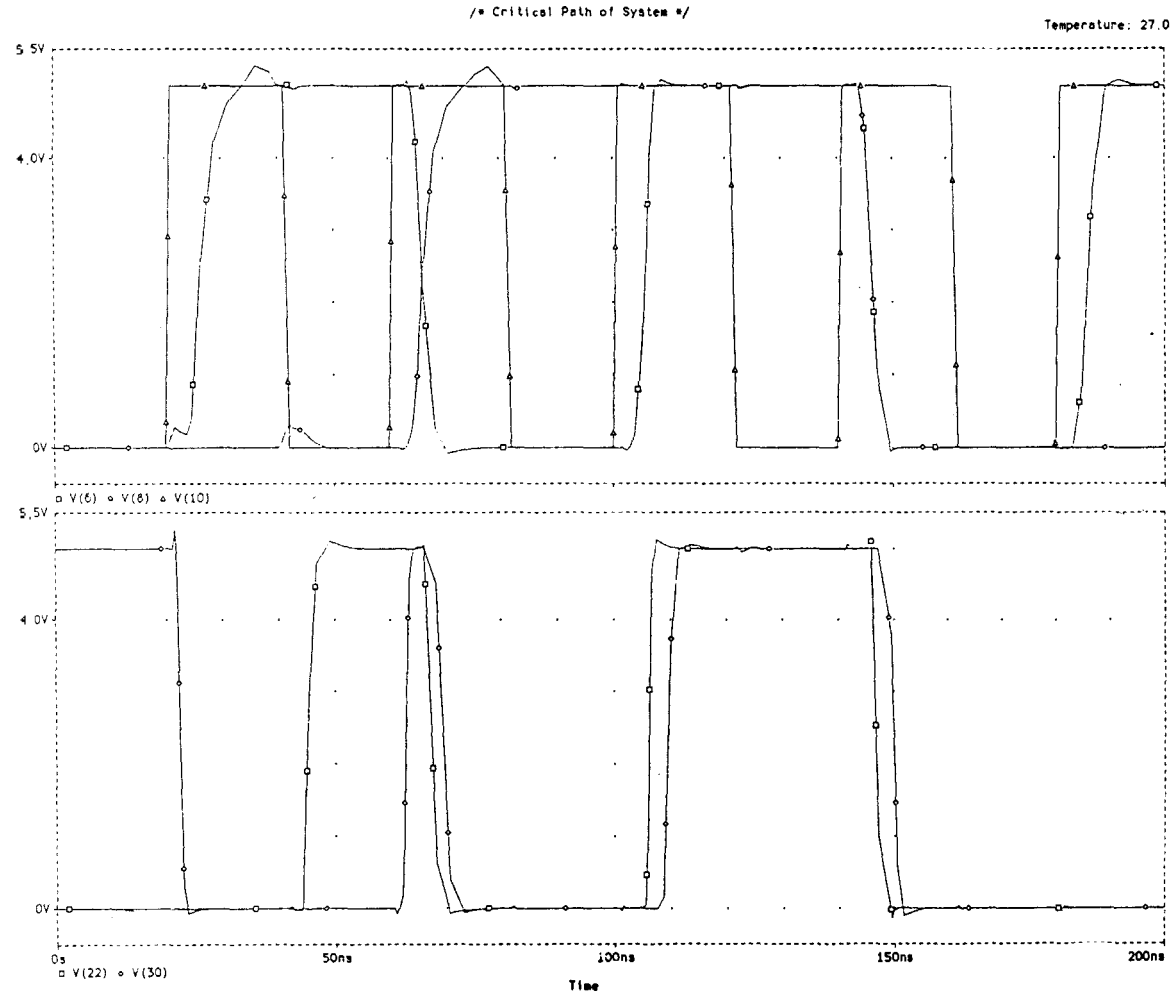
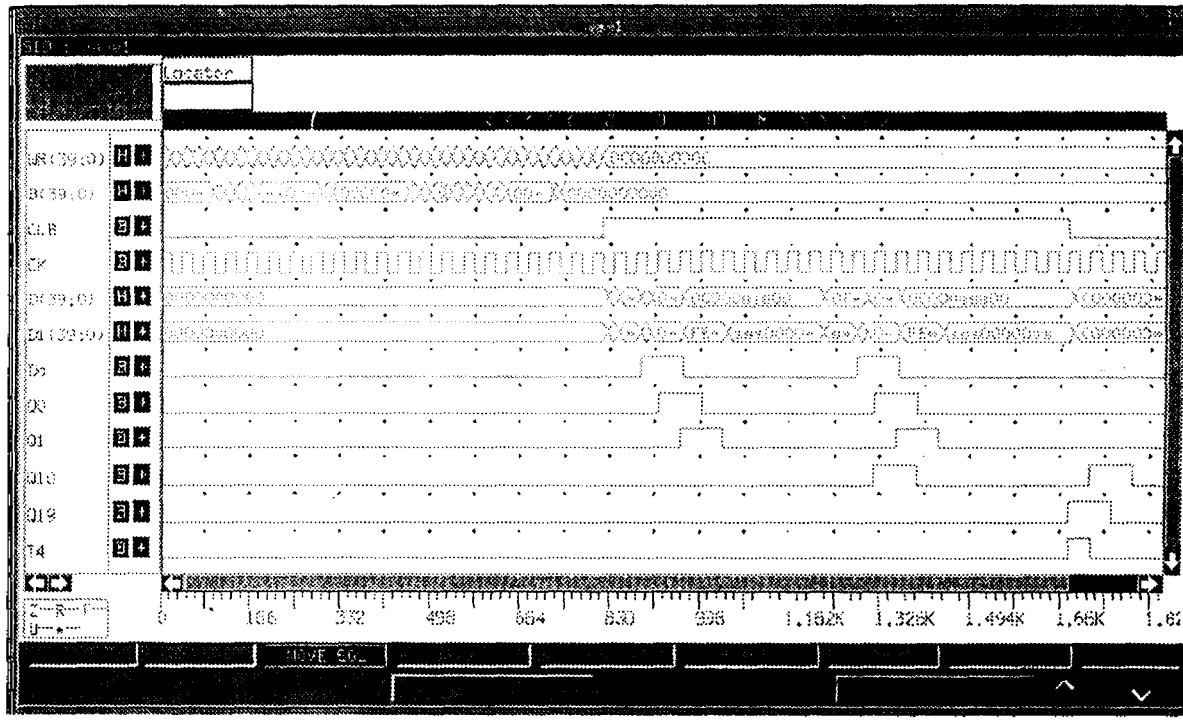


그림 15. SPICE 시뮬레이션 결과
Fig 15. SPICE simulation result of critical path



WR : write 신호　　B : 입력 데이타　　CLB : reset
CK : 클럭　　D : 매치된 결과　　D1 : 필터를 통해 나온 결과
Q(1-24) : 각 참조 테이블의 내용　　T(1~6) : 유형 분류 결과

그림 16. 시스템 타이밍도
Fig 16. System timing diagram

뒤 CAM에 저장된 데이타와의 비교를 거쳐 필터를 통해 그 결과가 참조 테이블에 저장되는 경로를 잡은 것이다. 여기서 bit1과 bit2는 임의의 데이타 입력 값이며 모든 블럭을 거쳐 최종적으로 참조 테이블에 저장되는 것을 보이고 있다. 그림 16은 본 논문에서 설계한 회로를 DAZIX의 DLS를 통해 얻은 시스템 타이밍도를 나타내고 있다. 타이밍도에서 알 수 있듯이 20개의 클럭을 사용하여 유형을 분류(T4)하였으며, 매 클럭 마다 참조 테이블에 쉬프트되면서 저장되는 것을 알 수 있다.

Ⅵ. 결 론

본 논문에서는 자소추출방법을 이용한 문자 인식 시스템에서 인식을 위한 전처리 과정인 유형 분류 과정을 CAM을 이용하여 칩으로 설계하였다. 기존의 자소추출 방법[8]을 이용한 문자 인식 시스템은 세선화 및 특징점 추출은 하드웨어화를 실현하였으나 유형분류는 소프트웨어에 의존하는 형태였다. 이는 전체적인 시스템의 속도 감소와 성능저하를 야기시킬 수 있다. 본 논문에서는 시스템의 성능을 저하를 막고 고속문자인식기의 구현을 위해 유형분류를 CAM을 이용하여 칩으로 설계하였다. CAM은 기존의 메모리와는 달리 자체내에서 비교 검색을 할 수가 있어 기존의 메모리를 사용하여 비교를 하는 것보다 월등히 빠른 수행 속도를 보인다. 또한 칩으로 설계함으로써 인식 단계의 부하를 줄일 수 있을 뿐만 아니라 문자 인식기 구현의 용이함을 기대할 수 있으며, 이식성이나 확장성에 있어서도 뛰어난 특성을 나타내고 있다. 본 논문에서는 최소한의 참조 패턴(20개)만을 사용하였으므로 보다 정밀하고 다양한 참조 패턴과 그에 대응하는 참조 테이블을 마련한다면 오분류율은 더욱 감소될 수 있으며, 이에 따른 동작 속도는 감소할 것이다. 설계된 회로에서는 제어 회로 없이 외부에서 받는 클럭 펄스(CLK)와 reset 신호로 모든 작업을 수행할 수 있어 제어를 간소화 시킬 수 있었다. 또한 설계한 회로는 소프트웨어와 비교하여 월등히 빠른 처리 속도를 나타내었다. 따라서, 기존의 바코드나 자기 코드를 대신할 수 있는 문자 인식기를 구현할 경우 보다 유용하게 사용될 수 있을 것으로 기대된다.

본 논문에서 사용한 CAM은 히스토그램(histogram)을 작성하는데에도 뛰어난 특성을 가지고 있어, 계속적인 연구를 통하여 이 회로에 부가적인 회로를 첨가한다면,

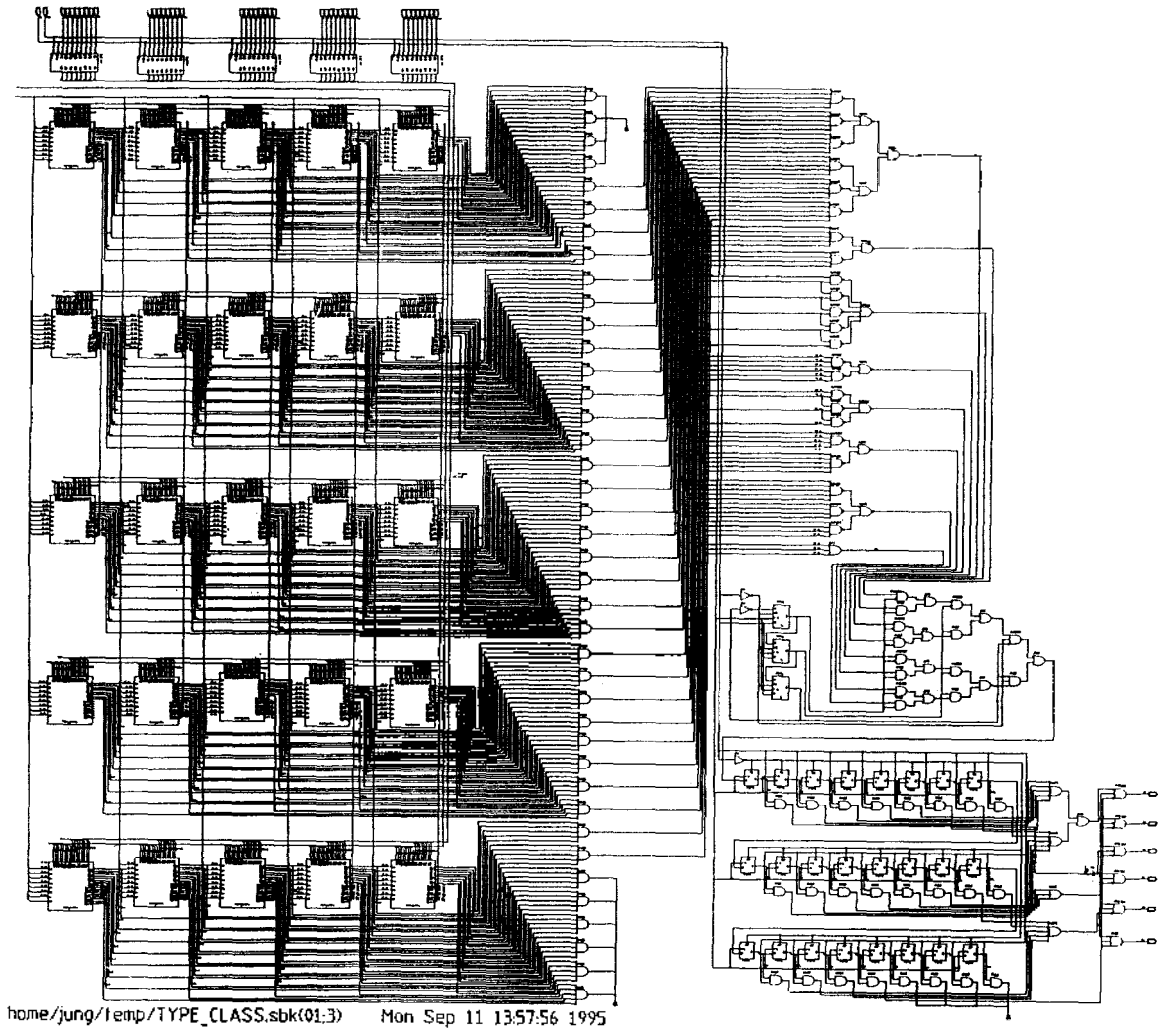


그림 17. 전체 시스템 회로도
Fig 17. System circuit diagram

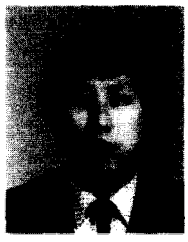
유형 분류와 함께 그 결과를 토대로 자소 분리까지 가능한 회로를 설계할 수 있을 것으로 사료된다.

참 고 문 헌

1. Lawrence Chisvin, R.James Dukworth, "Content-Addressable and Associative memory:Aternatives to the Ubiquitous RAM," IEEE Computer, pp. 51-64, July. 1989.
2. Anthony J.AcAuley and Charles J.Cotton, "A Selt-Testing Reconfigurable CAM," IEEE J.Solide State Circuits, Vol. 26, No. 3, pp. 257-261, March. 1991.
3. 박노경, 차균현, "High performance Content Addressable Mamory Design for pattern Inspection," proceeding of JTC-CSCC, pp. 507-510, 1995. 7
4. 박노경, 차균현, "연상 메모리 설계 및 제작에 관한 연구," 한국 통신학회 논문지, 제 16권, 제 2호, 1991.
5. 박노경, 차균현, "CARM(Content-Addressable Memory and Reentrant Memory)의 설계에 관한 연구," 한국 통신학회 논문지, 제 16권, 제 1호, 1988.
6. 강선미, 김덕진, 김혁구, "고속 처리를 위한 이진 영상 정규화 하드웨어의 설계 및 구현," 전자공학회, 제 31권, 제 5호, pp. 162-167, 1994. 5
7. 장석진, 강선미, 김혁구, 노우식, 김덕진, "자소 인식 신경망을 이용한 한글 문자 인식에 관한 연구," 전자공학회, 제 13권 제 1호, pp. 81-87, 1994. 1
8. 김민석, 손한웅, 최완수, 김수원, "자소추출방법을 이용한 고속 한글인식 시스템의 구현," 전자공학회, 제 29권, 제 6호, pp. 25-31, 1992. 6
9. 강선미, 김덕진, 김혁구, "고속 문자 인식을 위한 특징 추출용 칩의 구현," 전자공학회, 제 31권, 제 6호, pp. 104-110, 1994. 6
10. 이동현, 조원규, 양현승, 김진형, "신경망 모델을 사용한 문자의 형태 분류," 한국정보 통신공학회 봄 학술발표논문지, Vol. 16, No. 1, pp. 215-219, 1989.

11. 이주근, "한글 문자 인식에 관한 연구(IV)," 전자공학회, 제 9권, 제 4호, pp. 25-32, 1972. 9
12. 김혁구, 최선미, 김덕진, "고속 문자 인식을 위한 특징 추출용 칩의 구현," 전자공학회, 제 31권 B편, 제 6호, pp. 104-110, 1994. 6
13. 이성환, '문자 인식 이론과 실제 I, II,' 홍릉과학출판사, 1993.

▲박 노 경(Noh Kyung Park)



1984년 2월 : 고려대학교 전자공학과 (학사)
1986년 2월 : 고려대학교 전자공학과 (석사)
1990년 2월 : 고려대학교 전자공학과 (박사)
1988년 2월~현재 : 호서대학교 정보통신공학부 부교수
※주관심분야 : VLSI/CAD, 영상신호처리

▲구 창 모(Chang Mo Koo)

1992년 2월 : 호서대학교 정보통신공학과(학사)
1996년 2월 : 고려대학교 전자공학과 산업대학원(석사)
※주관심분야 : 문자 인식, 회로 및 시스템 설계

▲정 장 원(Chang Won Jeong)

1995년 2월 : 호서대학교 정보통신공학과(학사)
1995년~현재 : 호서대학교 정보통신공학과 석사과정 재학중
※주관심분야 : VLSI 신호 처리, 회로 및 시스템 설계