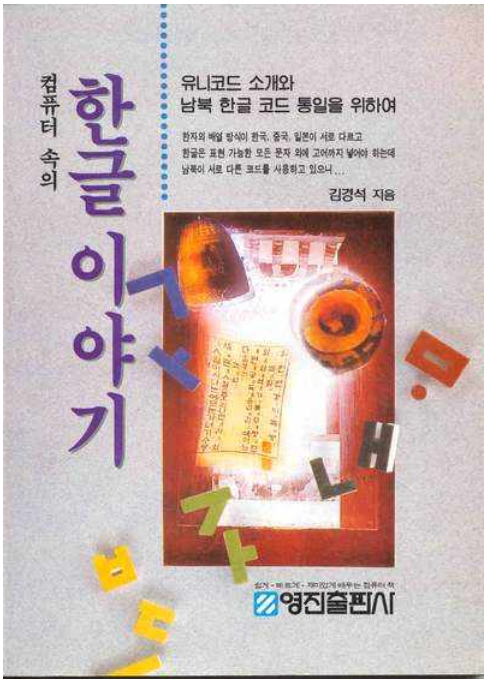


컴퓨터 속의 한글 이야기

(첫째 보따리)

김 경석

영진 출판사, 1995.03.

http://asadal.pusan.ac.kr/~gimgs0/book/hgiyagi1.html

제 25 장
옛한글 문서 편집기 나랏 말씀 개발 소개

글쓴이는 대학원생들과 같이, ISO 10646-1 에 있는 첫가끝 한글 부호계 238 글자를 써서 옛한글까지 제대로 지원하는 옛한글 문서 편집기를 개발하여 일반에게 거저 공개하고 있는데, 이 편집기의 개발에 대해서 소개하겠다. 옛한글 문서 편집기는, 지금까지 개발된 문서 편집기에서는 편집할 수 없었던, 옛한글을 완벽하게 입력하고, 이를 문서로 작성하여 파일로 저장하고 인쇄할 수 있다.

옛한글 문서 편집기에서 쓴 옛한글 자판, 옛한글 입력기 및 출력기, 옛한글 문서 편집기의 주요 기능, 앞으로 개선되어야 할 사항에 대해서 살펴 본 뒤, 프로그램의 구성, 편집기의 개발 환경, 필요한 사용자 환경을 보겠다.

25.1 옛한글 자판의 개발

지금까지 사용하고 있는 한글 자판은 요즘한글을 입력할 수 있도록 설계된 자판이다. 따라서 옛한글 문서 편집기에서는 기존의 요즘한글뿐만 아니라 옛한글 글자들도 입력할 수 있는 옛한글 자판이 필요하다, 여기에는 옛한글 문서 편집기에서 사용하는 국제 표준 한글 부호계의 특성도 고려되는 것이 바람직하다고 생각된다. 본 개발에서는 기존의 세벌식 자판을 기본으로 사용하여 옛한글 글자를 입력할 수 있는 자판을 개발하였다.

25.2 글쇠 위치의 선정

자판의 설계에서 가장 중요한 문제는 입력에 필요한 글쇠들의 위치를 결정하는 문제이다. 겹글자는 그 겹글자를 이루는 홑글자를 차례대로 치도록 했으므로, 옛 홑글자만 자판에 배열하면 된다. 아래의 표에서 알 수 있듯이, 첫소리 홑글자 9개, 가운뎃소리 홑글자 1개 , 끝소리 홑글자 3개 등 모두 13 개의 홑글자와 2 개의 방점을 배열하면 되는데, 문제는 현재 사용하고 있는 요즘 한글 자판에 어떻게 이 15 개의 글자를 더하느냐는 것이다.

표 25-1. 국제 표준 한글 부호계에 들어 있는 옛한글 글자 수

	홑글자	겹글자	모두
첫소리 글자	9	62	71
가운뎃소리 글자	1	44	45
끝소리 글자	3	52	55
모두	13	158	171

글쇠의 자리를 정할 때 고려해야 할 사항이 여러 가지이겠지만, 가장 중요한 점은 입력을 빨리 할 수 있는가이다. 그런데 옛한글 자판의 경우에는 좀 특수한 면이 있다. 평범한 사람은 많이 쓰지 않을 것이며, 또한 거의 모든 사람들이 이미 요즘 한글 자판에는 익숙해 있을 것이다. 따라서 가능하다면 사용자가 빨리 배워서 쓸 수 있는, 다시 말해서 외우기 쉬운 자판을 만드는 것이 옛한글 자판의 경우에는 꽤 중요하다고 본다. 이런 관점에서, 이번 개발이 옛한글 처리에 대한 거의 첫 시도인 만큼 사용자가 외우기 쉽게 글쇠의 위치를 선정하였다.

구체적으로, 기존 세벌식 사용자들이 많이 쓰고 있는 공 병우 세벌식 390 자판을 기본으로 하여, 거기에 옛한글 글자들을 더했다. 현재 쓰고 있는 공 병우 390 자판에는 모든 글쇠에 이미 글자가 배당되어 있기 때문에, 옛한글 글자들은 윗글자쇠 (SHIFT 키) 와의 조합으로 칠 수 있게 하였는데, 보기를 들면 '옛 이응 ㅇ(꼭지 달린 이응)' 은 SHIFT + 'ㅇ'로 하였다.

옛 겹글자들은 따로 글쇠를 두지 않고, 공 병우 세벌식 390 자판에서와 같이, 겹글자를 이루는 각 홑글자를 하나씩 쳐서 입력하는 방식을 사용한다. 보기를 들어, 옛한글의 첫소리 글자 'ㅂㅅㄱ'은 자판에서 첫소리 글자 'ㅂ' + 'ㅅ' + 'ㄱ' 과 같이 친다. 옛 가운뎃소리 겹글자나 옛 끝소리 겹글자도 마찬가지이다.

아래 그림은 옛한글 문서 편집기 나랏 말씀에서 제공되는 세벌식 옛한글 자판이다.

그림 25-1. 세벌식 옛한글 자판 (93-12-31)

25.3 옛한글 입력기의 설계 및 구현

옛한글 입력기는 사용자가 옛한글 자판에서 친 글자들을 조합하여 이를 소리마디로 만들어준다. 입력기는 자판이 두벌식인지 세벌식인지에 따라 매우 달라지는데, 세벌식 자판을 사용하는 경우는 첫소리 글자와 끝소리 글자의 위치가 다르기 때문에, 입력기에서 이를 구분해 줄 필요가 없다. 이에 반하여 두벌식 자판을 사용할 경우는 첫소리 글자와 끝소리 글자의 글쇠 위치가 같기 때문에, 이를 입력기에서 처리해 주어야 한다. 본 문서 편집기에서는 세벌식 자판을 기본으로 하기 때문에 입력기 설계가 아주 간단하다.

25.3.1 입력 오토마타의 설계

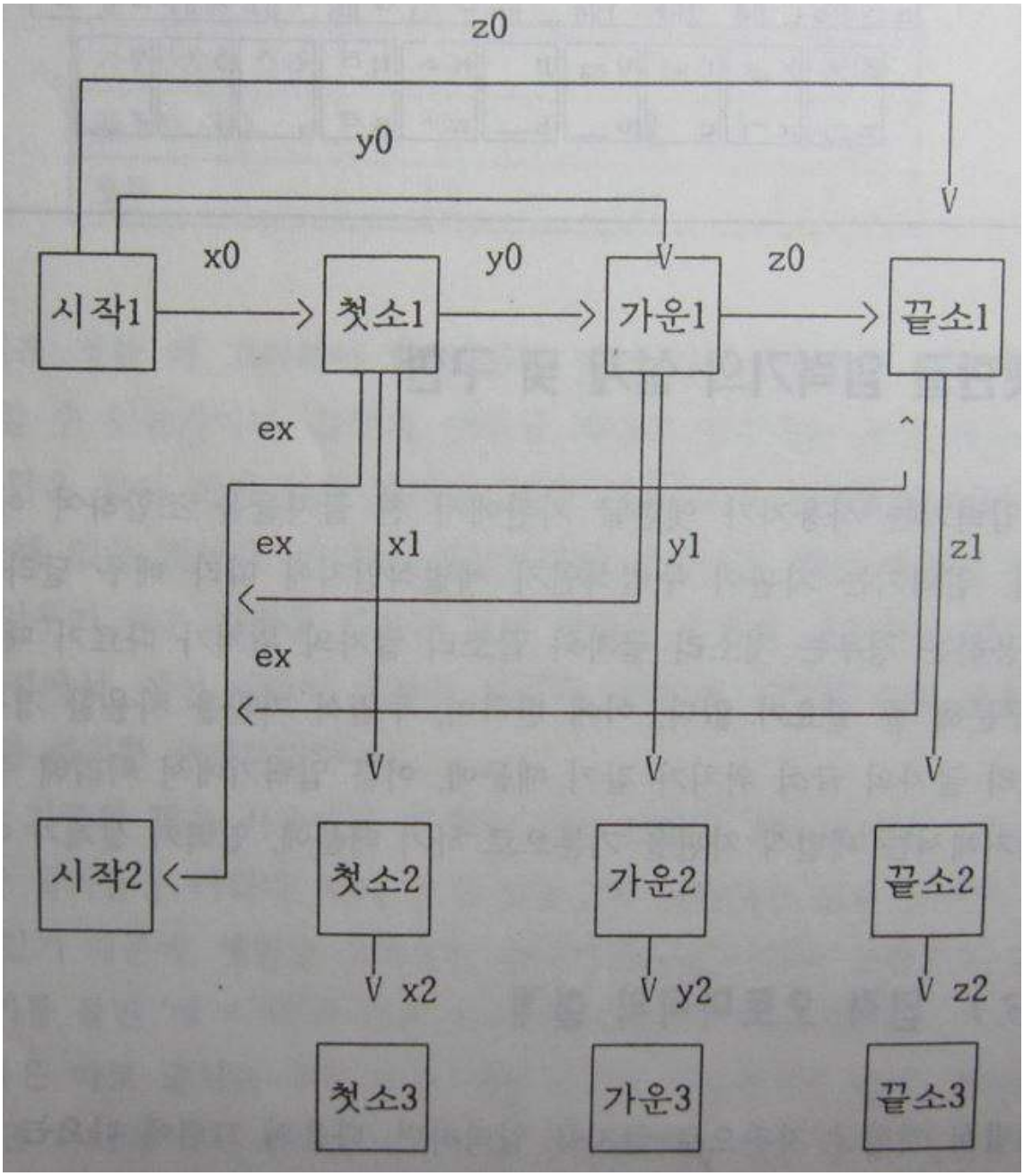
세벌식 옛한글 자판으로 글자를 입력하면, 아래의 그림에 나오는 입력 오토마타가 동작하여 소리마디로 만들어 준다.

세벌식을 사용하는 경우는 사실 입력 오토마타라는 말이 거의 뜻이 없다. 세벌식에서는 첫소리 글자와 끝소리 글자가 다르게 들어 오기 때문에, 그냥 들어오는 대로 처리하면 된다. 그에 비해서, 두벌식에서는 닿소리 글자로 들어온 것을 첫소리 글자와 끝소리 글자로 나누어야 하기 때문에 입력 오토마타라는 말이 나오게 되었는데, 이런 점에서 보더라도 두벌식은 한글의 특성에 맞지 않는 것 같다.

세벌식 한글 입력 오토마타에서는 주로 겹글자와 한글 입력상태에서 영어 입력상태로의 전환들의 상황 변화와 백 스페이스 등의 특수 키들이 입력되었을 경우 이를 적절히 처리할 수 있도록 해 준다.

특히 옛 겹글자가 많기 때문에 옛한글 입력 오토마타에서 검사해야 할 겹글자가 많으며, 또한 옛 홑글자의 입력도 고려해야 한다.

그림 25-2. 세벌식 옛한글 입력 오토마타



< 상태설명 >

첫소1: 첫소리 글자 입력상태
첫소2: 첫소리 겹글자 상태
첫소3: 첫소리 세 겹글자 상태
(예, 'ㅂ+ㅅ+ㄱ')

가운1: 가운뎃소리 글자 입력상태
가운2: 가운뎃소리 겹글자 상태
가운3: 가운뎃소리 세 겹글자 상태
(예, 'ㅜ+ㅓ+ㅡ')

< 입력 설명 >

x0 : 첫소리 글자
x1 : 겹글자가 가능한 글자
x2 : 세 겹글자가 가능한 입력

y0 : 가운뎃소리 글자
y1 : 겹글자가 가능한 글자
y2 : 세 겹글자가 가능한 글자

끝소1: 끝소리글자 글자 입력상태　　z0 : 끝소리 글자
끝소2: 끝소리글자 겹글자 상태　　z1 : 겹글자가 가능한 글자
끝소3: 끝소리글자 세 겹글자 상태　　z2 : 세 겹글자가 가능한 글자
　　(예, 'ㄹ+ㅁ+ㅂ')

시작2: 입력을 유지하면서　　ex : 그 외의 글자
　　시작1 상태로 분기

25.3.2 입력기의 구현

가. 입력 오토마타의 구현

입력 오토마타를 프로그램으로 구현하는 방법은 여러 가지가 있으나 본 문서 편집기에서는 비교문을 이용하여 구현하였다.

사용자가 글쇠를 입력할 때의 입력 모드를 검사하여, 영문 모드일 경우는 해당하는 영문 코드를 생성하고, 한글 모드일 경우는 별도의 한글 처리 루틴에서 한글 입력 오토마타의 상태와 입력을 고려하여 상태 전이 및 코드를 생성한다. 특히 한글 입력에서는 겹글자를 이룰 수 있는지에 대한 검사를 하여야 하는데, 이는 별도의 루틴에서 검사하도록 한다. 아래는 입력 오토마타 프로그램을 간단히 나타낸 것이다.

```
        :
    if( 입력 모드가 영문이면 ) {
        영어 코드 생성
    }
    else { // 한글 입력 모드
        입력된 글쇠에 대한 코드 생성
        switch ( han_state ) {
            case START :
                        if(첫소리 글자이면 ) {
                            han_state = SylInit
                        }
                        else if(가운뎃 소리 글자이면 ) {
                            han_state = SylPeak
                        }
                        else { // 끝소리 글자
                           han_state = SylFin
                        }
                        break;
            case SylInit:
```

```
                    if(첫소리 글자이면 )
                        겹글자가 가능한지 검사
                    else 다음 상태로 전이
        case SylPeak:
                    if( 가운뎃소리 글자이면 )
                        겹글자가 가능한지 검사
                    else 다음 상태로 전이
        case SylFin:
                    if( 끝소리 글자이면 )
                        겹글자가 가능한지 검사
                    else {
                        소리마디 완성
                        다음 상태로 전이
                    }
          :
          :
```

나. 겹글자의 처리

옛한글의 겹글자는 첫소리 겹글자 62개, 가운뎃소리 겹글자 45개, 끝소리 겹글자 52개 등 모두 158개이다. 여기에 요즘한글 겹글자를 더하면 처리해야 할 겹글자는 아주 많다.

겹글자의 처리는 입력 오토마타에서 겹글자 처리 루틴을 호출하는 형태로 처리되는데, 겹글자가 가능한지를 검사하기 위해서 아래와 같이 구성된 겹글자 테이블이 사용된다.

기본 글자	겹글자가 가능한 글자	생성되는 겹글자 코드

```
struct {
        int b ;
        int code ;
        int out ;
} table [size_SylInit]
```

첫소리 글자 입력 상태에서 다시 첫소리 글자가 입력되면 입력 오토마타에서 겹글자 처리 루틴을 호출하고, 여기에서 겹글자 테이블을 검색하여 겹글자로 만들어질 수 있는지를 검사하게 된다. 가운뎃소리 글자나 끝소리 글자도 마찬가지이다.

겹글자가 가능한 경우는 겹글자 테이블에 저장되어 있는 겹글자에 해당하는

코드를 부여한 뒤 다음 입력을 처리하게 되며, 그렇지 않은 경우는 이미 입력된 부분으로 소리마디를 완성시킨 뒤 새로운 상태로 전이하게 된다.

라. 불완전한 소리마디의 처리

입력되는 소리마디가 첫소리-가운뎃소리 글자 또는 첫소리-가운뎃소리-끝소리 글자로 이루어지는 소리마디를 완전한 소리마디라고 하고 그 밖의 것을 불완전한 소리마디라고 정의한다. 이러한 불완전한 소리마디를 나타내는 방법은 국제 표준 한글 부호계에 이미 정의되어 있다. 그 내용을 살펴보면 다음과 같다.

- 첫소리 글자뿐 : 첫소리 글자, 가운뎃소리 채움 글자
- 가운뎃소리 글자뿐 : 첫소리 채움 글자, 가운뎃소리 글자
- 끝소리 글자뿐 : 첫소리 채움 글자, 끝소리 글자

(참고. 나랏 말씀을 개발할 때, 그리고 이 원고를 쓸 때인 1995 년에는 1993 년에 출판된 국제 표준에 끝소리 글자뿐인 때 이렇게 나타내었다.)

본 문서 편집기에서는 불완전한 소리마디 처리를 위의 방식에 따랐는데, 한 가지 예외가 있다. 끝소리 글자뿐일 때는, 한글 출력 루틴을 간단하게 구현하기 위해서 아래와 같이 처리 하였다.

- 끝소리 글자뿐 : 첫소리 채움 글자, 가운뎃소리 채움 글자, 끝소리 글자

(참고. 1993 년 판이 그 뒤에 개정되어, 원고를 고쳐 쓰는 때인 2021 년에는 국제 표준에 끝소리 글자뿐인 때 이렇게 나타낸다.)

25.4 옛한글 출력기의 설계와 구현

25.4.1 옛한글 출력기의 설계 환경

옛한글 문서 편집기는 VGA 이상의 그래픽 카드를 기본 사양으로 하기 때문에. 출력기도 이 환경에서 동작하도록 구현되었다. 또한 16x16 글꼴로 옛 겹글자를 화면에 나타내기 어렵기 때문에, 본 문서 편집기에서는 24x24 font를 사용하였다. 조합방식으로 한글을 처리할 경우, 소리마디가 화면에 표시될 때의 모양을 보기 좋게 하기 위해서 여러 벌의 글꼴을 사용하는 것이 일반적이나, 본 개발에서는 주어진 시간 안에 개발을 끝내기 위해서, 첫소리, 가운뎃소리, 끝소리 글자 각 1벌씩 해서, 모두 3 벌의 글꼴을 사용하여 구현하였다.

한글 처리에 있어서 출력 방식을 결정하는 또 하나의 문제는 한글 처리를 위한 별도의 하드웨어를 사용하느냐 않느냐는 점이다. 현재 많이 쓰는 문서 편집기는 하드웨어의 도움 없이 한글 출력을 자체적으로 처리하는 것이 대부분인데, 이러한

방법은 사용자가 별도로 하드웨어를 사지 않아도 프로그램을 사용할 수 있다는 장점이 있기 때문에, 본 옛한글 문서 편집기도 하드웨어를 쓰지 않기로 하였다.

옛한글 출력기는 다음과 같은 환경에서 쓸 수 있도록 만들었다.

그래픽 카드 : VGA 이상 (640x400 mode)
사용 글꼴 : 한글 (옛한글 포함) 24x24 비트맵 글꼴
영어 12x24 비트맵 글꼴
처리방법 : 소프트웨어 처리

25.4.2 옛한글 출력기의 구현

가. 출력루틴의 설계

출력 루틴의 설계에서 가장 중요한 부분은 비트맵 글꼴을 화면에 나타나도록 하는 부분이다. 이 부분은 글꼴의 크기, 정확히 말하면 글꼴을 바이트 단위로 처리할 수 있는지에 따라서 구현이 달라진다.

본 문서 편집기에서 사용하는 글꼴은, 한글일 경우 24x24 크기를 가지기 때문에 24 픽셀 한 줄을 3 바이트로 나누어 처리할 수 있지만, 아스키 문자일 경우는 12 픽셀 한 줄을 바이트 단위로 정확히 나눌 수 없기 때문에 바이트 단위의 처리가 불가능하다. 따라서 전체적인 화면을 구성하는 입장에서 보면, 한 화면에는 한글과 영어가 동시에 섞여 나올 수 있기 때문에, 화면의 글꼴을 바이트 단위로 처리하지 못한다. 이러한 문제점을 해결하기 위해서, 글꼴 처리는 가장 기본적인 단위인 픽셀 단위로 구현하였다. 이 방법은 출력 속도의 문제점이 지적될 수 있으나, 글꼴의 크기를 바꿀 때 비교적 쉽게 루틴을 재 구성할 수 있다는 장점이 있기 때문에, 이번 첫 시도에서는 이 방법을 사용하였다.

아래는 출력 루틴에서 가장 기본적인 부분인 한 글자를 화면에 출력하는 루틴을 간략화한 것이다.

```
void Putdot_han24( int x, int y, unsigned char *buf, int color)
{
     int i, j, k;
     char temp, mask[8] = { 0x80, 0x40, 0x20, 0x10, 0x8, 0x4, 0x2, 0x1} ;
                    :
     for (j = 0 ; j < 12 ; j++ ){
          for (i = 0 ; i < 3 ; i++, buf++) {
               for (k = 0 ; k < 8 ; k ++)
                    if (mask[k] & *buf) {
                       /* 한 픽셀을 화면에 출력 */
                       hgPlotXy(x+i*8+k, y+j, color);
                    }
```

```
                    :
                    :
}
```

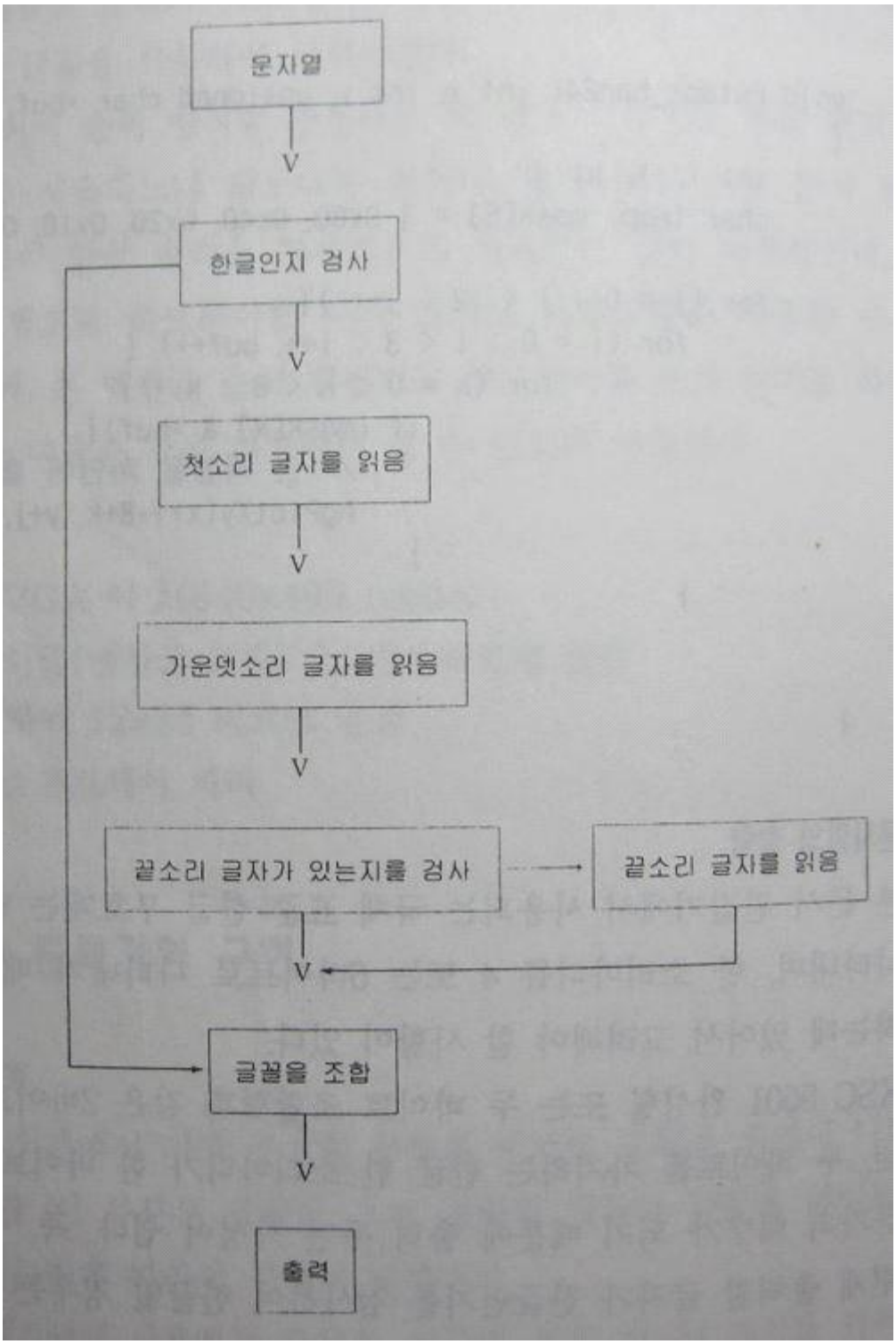
나. 문자열의 출력

본 문서 편집기에서 사용되는 국제 표준 한글 부호계는 한 글자를 2 바이트로 나타내며, 한 소리마디를 4 또는 6 바이트로 나타내기 때문에 문자열을 출력하는데 있어서 고려해야 할 사항이 있다.

KSC 5601 완성형 또는 두 바이트 조합형과 같은 2 바이트 한글 부호계를 쓰면, 두 바이트를 차지하는 한글 한 소리마디가 한 바이트를 차지하는 아스키 문자의 배수가 되기 때문에 출력 루틴 작성이 쉽다. 즉 주어지는 문자열에서 현재 출력할 글자가 한글인지를 검사하여 한글일 경우는 다음 바이트를 강제로 읽어서 소리마디를 구성한 다음 글자를 출력하면 된다.

그러나 국제표준 한글 부호계를 사용할 경우, 한글 한 소리마디가 끝소리 글자가 있느냐 없느냐에 따라, 6 또는 4 바이트를 차지하여, 영어 한 글자의 일정한 배수가 되지 않기 때문에 두 바이트 한글 부호계에서 쓰는 방법을 쓸 수 없다. 따라서 문자열을 출력할 때 한글이 나오면, 끝소리 글자가 있는지를 검사해야 하며 끝소리 글자가 있을 경우에는 2바이트를 더 읽은 다음 각 글자의 글꼴을 조합하여 한 소리마리에 해당하는 이미지를 만들어 화면에 출력해야 한다. 아래 그림은 문자열 출력에 대한 흐름도이다.

그림 25-3. 문자열 출력을 위한 흐름도



25.5 문서 편집 기능의 구현

본 문서 편집기의 편집 기능들은 사용자들이 되도록 쉽게 옛한글 문서 편집기를 사용할 수 있도록, 아래아 한글 2.0 문서 편집기와 사용법과 거의 같도록

구현했다. 이번 개발에서는 기본적인 문서 편집 기능을 지원하는 것을 목표로 하였으며, 이 부분은 앞으로 많은 기능을 더 넣어야 할 것으로 본다.

25.5.1 옛한글 문서 편집기의 주요 기능

본 문서 편집기의 주요 기능을 살펴보면 다음과 같다.

가. 도움말 기능

옛한글 문서 편집기를 사용할 때 필요한 자판 배열과 단축 키들을 사용자에게 알려준다.

자판 배열 - 옛한글 문서 편집기에서 제공하는 글쇠의 위치들을 출력한다

도움말 - 옛한글 문서 편집기에서 사용할 수 있는 단축키의 종류와 기능들을 보여준다

나. 파일 처리

파일 읽기 - 이미 작성된 파일을 옛한글 문서 편집기로 읽어 들인다.
파일 쓰기 - 작성한 파일을 저장한다.
새이름으로 - 파일을 새로운 이름으로 재 저장한다.
경로 바꾸기 - 작업하고 있는 디렉토리를 바꾼다.
도스 명령어 - 문서 작성 중에 도스로 잠깐 나간다.
끝내기 - 작업을 마친다.

다. 출력 기능

옛한글 문서 편집기에서는 간단한 출력 기능을 제공한다.

프린터는 PCL3 (printer control language 3)를 지원하는 기종을 대상으로 하였는데, 사용할 수 있는 프린터는 계속 추가할 예정이다. PCL3 를 지원하는 프린터로는 HP Laser Jet, HP Desk Jet (PCL3 를 지원하는), 그리고 HP Laser Jet 을 emulate 하는 여러 프린터 등이다.

프린터 설정 - 프린터의 기종을 설정한다 (현재는 PCL3 를 지원하는 프린터 한 가지만 쓸 수 있음)
여백주기 - 프린트할 때 여백의 길이를 결정한다.
프린트 - 문서를 프린트한다.

25.6 앞으로 개발 방향

이번의 옛한글 문서 편집기 개발의 가장 중요한 목표는 국제 표준 한글 부호계를 시험 운용해 보고 이를 사용하여 옛한글 문서 편집기를 개발할 수 있는 가능성을 검증하는 것이었다. 따라서 글꼴을 예쁘게 하거나, 다른 문서 편집기에서 되는 기능을 다 수용할 수는 없었다.

앞으로 지속적인 개발을 통해, 더 나은 옛한글 문서 편집 기능을 제공해야 할 것으로 보는데, 주요한 점만 몇 가지 들면 다음과 같다.

1) 유닉스의 X-window 환경에서 쓸 수 있는 옛한글 문서 편집기를 개발하여, 유닉스 환경에서도 쓸 수 있도록 해야 할 것이다.
2) 옛한글 글꼴 개발은 옛한글 문서 편집기를 발전시켜 나가는데 중요한 문제이다. 요즘 많이 쓰고 있는 바탕체 글꼴에 옛한글 글자를 수용하는 것도 하나의 좋은 방안이라고 본다.
3) 옛한글 자판은 아직 정립되지 않았으며, 능률적인 자판을 개발하고 시험해 볼 수 있도록 사용자가 옛한글 자판을 정의하여 쓸 수 있도록 지원해야 할 것이다.

25.1 프로그램의 구성

옛한글 문서 편집기 프로그램은 옛한글 문서 편집기가 사용할 그래픽 환경을 만들어 주기 위한 부분과 옛한글 문서 편집기의 실제 처리 부분으로 크게 나눌 수 있다.

옛한글 편집기 부분은 모두 9개의 C 프로그램과 2개의 헤더 파일로 구성되어 있는데, 모두 약 4,000 줄이다. 아래의 표는 각 C 프로그램에 대한 간략한 설명이다.

표 25-2. 옛한글 문서 편집기를 이루는 C 프로그램

파일 이름	기능
ohwp.c	옛한글 문서 편집기의 편집 기능을 구현
hanlib.c	옛한글을 처리하는데 필요한 기능들을 정의
myutil.c	문서 편집기에 필요한 여러 가지 기능들을 정의
screen.c	화면 처리에 대한 함수들로 구성
gwindow.c	옛한글 문서 편집기에서 사용하는 윈도우를 처리
kbd.c	옛한글 자판을 화면에 출력
function.c	문서 편집기의 메뉴에 대한 처리

25.8 옛한글 문서 편집기 개발 및 사용 환경

본 문서 편집기는 IBM-PC 486에서 “C" 언어를 사용하여 구현하였다. 그러나 한글 출력 루틴에 필요한 그래픽 처리 부분은 어셈블리 언어가 사용되었다.

본 문서 편집기는 그래픽 환경에서 동작하기 때문에 개발 과정에서도 VGA 그래픽 카드를 사용하였다. 자세한 개발 환경은 다음과 같다.

. 운영체제 : MS-DOS version 5.0

. 사용 피씨 : IBM-PC 486 DX
. 그래픽환경 : VGA 그래픽 카드,
256 칼라 이상을 지원하는 칼라 모니터
. 사용언어 : C 언어
. 컴파일러 : Borland C/C++ ver 3.1

옛한글 문서 편집기를 사용할 사용자는 VGA 그래픽 카드를 장착한 IBM-PC XT이상의 컴퓨터와 MS-DOS version 3.XX이상의 운영체제를 준비하면 본 문서 편집기를 운용할 수 있지만, 가장 좋은 성능을 위해서는 아래와 같은 환경을 권한다.

. MS-DOS version 3.XX 이상
. IBM PC-386 이상
. 1 MB 이상의 주기억 장치
. VGA 이상의 그래픽 카드
. 하드 디스크를 장착한 피씨

25.9 맺음말

본 개발의 가장 중요한 목표는, 과연 국제 표준 한글 부호계에 있는 새로운 첫가끝 조합형을 실제로 구현할 수 있는지를 알아 보는 것이었으며, 그 시험 대상으로 상당히 복잡한 옛한글 문서 편집기를 골랐다. 개발이 끝난 뒤 평가해 보건데, 새로운 첫가끝 조합형은 충분히 구현할 수 있다는 것이다. 주어진 시간 안에 개발을 마치기 위해서 가장 기본적인 편집 기능만 넣었으며, 글씨꼴은 그렇게 예쁘지 않다. 프로그램은 c 언어를 썼으며, 모두 약 4,000 줄로 되어 있다.

자판은 세벌식 옛한글 자판을 개발하여 썼는데, 능률적이고 구현하기 쉽다는 점 때문에 세벌식을 썼다. 공 병우 세벌식 390 요즘 한글 자판에 옛 홑글자 13 개와 방점 2 개, 모두 15 개의 글자를 더 넣었다. 옛 겹글자는 겹글자를 이루는 홑글자를 차례대로 치면 된다. 앞으로 두벌식 옛한글 자판에 대해서도 개발이 필요하다.

옛한글 문서 편집기는 VGA 카드가 있는 피씨에서 쓸 수 있으며, 하드웨어를 따로 살 필요가 없다. 프린터로는 HP 또는 HP 를 emulate 하는 프린터 (정확하게 말하면 PCL3 를 지원하는 프린터) 를 쓸 수 있다.

앞으로, 유닉스의 X-window 에서 쓸 수 있는 옛한글 문서 편집기를 개발하여, 유닉스에서도 옛한글을 처리할 수 있도록 하는 것이 한글 사전 개발이나 한글 자료의 데이터베이스화를 위해서 중요하다고 본다. 또한 옛한글 글꼴을 개발하여 보급하는 작업도 이루어져야 하겠다.

(2021.12.03.에 고쳐 씀)