ko.TEX 이전의 한글 LATEX
역사적 탐구와 체험

Nova de Hi

2022년 4월

차 례

1

서론

이 작업물은 2022년 현재 시점에서 한글 라텍의 발전과정을 돌이켜보고자 하는 사람에게 당시 한글 라텍 작업 환경을 체험할 수 있도록 당시 사용하던 기계를 가상 머신으로 구현하고 거기에 각 TEX 시스템과 한글 환경을 재현하여 제공한다. 이를 '체험판'이라 부르겠다.

(1) 한글 LATEX만을 문제삼을 것이므로 예컨대 plainTEX에서의 한글 구현 등은 제외하였다.

(2) 가상 기계는 Windows 운영체제에서 실행되는 것으로 제한하였다. 그러나 ova 가상 시스템이나 DOSBOX 용 파일 등은 다른 운영체제 하에서도 활용할 수 있을 것이다. SheepShaver 체험 파일로 제공되는 것은 압축 파일에서 필요한 것을 알맞게 재배치하여 시험할 수 있다.

(3) 2000년 이전까지만 해도 TEX 사용의 가장 어려운 곳이 '설치'였다. 이 문제가 거의 해결된 지금 "그랬다"고 한들 공감을 얻기 힘들겠지만 실상이 그러했고 지금도 당시의 '설치' 상황을 재현하는 것은 매우 시간이 걸리고 까다로운 일이다. 이 체험판은 '설치'의 괴로움을 체험하게 하려는 목적이 아니기 때문에 모든 설정이 다 완비된 상태로 제공한다.

이를 체험하면서 각 시기 한글LATEX의 상황과 해결해야 할 문제가 무엇이었는지, 그리고 이를 어떻게 해결해왔는지를 점검해보는 것이 이 글의 목적이다.

예전의 TEX 사정을 되돌아보는 것은 고고학적 취미에만 그치는 것은 아닐 것이다. 역사적인 가치도 없지 않을 것이고 한글 LATEX의 발전에 기여한 여러 분들의 노고를 돌이켜보는 점에서도 의미가 없지 않으리라 생각한다. 그러나 이 작업물의 가장 큰 목적은 '재미'이다. 잠깐의 즐거움이 되기를 바란다.

이 중의 일부 가상 시스템은 KTUG 게시판에 이미 소개한 바가 있다 (부록 D 참조).

2

선사시대: yahTeX과 WinLaTeX

'선사'라고 한 것은 조크이다. 대체로 1990년경부터 1995년까지에 해당하는 이야기다.

2.1 EmTeX과 yahTeX

원래 메인 프레임워크 급의 컴퓨터에 터미널로 연결해서 사용하던 TEX이 개인용 컴퓨터로 들어오는 데 가장 크게 기여한 것은 εmTeX이라는 DOS용 프로그램이었다. TEX 대중화의 주역이었다고 해도 무방할 것이다.

그리고 이 새로운 시스템에 흥미를 보이기 시작한 대학생 대학원생들이 중심이 되어 한글 구현 문제의 해결책이 모색되기 시작했다. 그 가운데 나름 성공적인 결과를 보여준 것은 1991년경의 yahTeX이었다. (yahTeX이라는 이름은 "yet another hangul tex"에서 온 것이라 한다.) 이외에도 몇 종류의 한글 TEX 구현이 있었던 것으로 보이나 yahTeX만큼 지속적 개발이 이루어지지 않았다. yahTeX은 1992년의 0.62버전으로 개발이 중단된다.

처음에 이 시스템은 상용조합형 한글을 처리했다. 그러다가 0.6버전부터 완성형 한글을 사용하게 되었는데 지금 흔적으로나마 남아 있는 것 중에 상용조합형 한글을 사용하는 0.5 이전 버전은 찾을 수 없다.

특징을 요약한다.

(1) MS-DOS, EmTeX에 부가 설치.

(2) 300dpi bitmap 글꼴 제공.

(3) LATEX version 2.09. TEX 3.141

(4) 한글을 전처리하는 방식.

이 시스템을 '선사' 취급하는 이유는 전처리 방식을 사용했음에도 불구하고 전처리 프로그램 자체를 DOS용 binary(exe)로만 제공하였다는 점 때문이다. 요컨대, 소스를 공개하지 않았다. 그러니 지금 와서는 아무 짝에 쓸모없는 물건이 되고 말았다.

2.2 yahTeX을 DOSBOX에서 실행해보기

εmTeX은 설치가 까다롭기로 유명했다. 그렇게 된 이유는 너무 인기있는 프로그램이다보니 여러 사람이 이런저런 기여를 하기 시작하면서 꼬여버린 게 많았기 때문이고 하드웨어에 따라 설정값이 달라졌기 때문이다. 예컨대 EmTeX base 시스템이 성공적으로 설치되었다고

해도 곧 DOS의 메모리 부족에 시달렸기 때문에 Extended memory, Expanded memory를 설정해야 했으며 TEX 사용 메모리 설정을 확장하기 위해 BigTEX이라는 것을 또 설정해줘야 했으며…….

1992-yahtex-emtex_dosbox.zip 파일은 DOSBOX에서 EmTeX과 yahTeX을 체험해볼 수 있도록 구성된 파일 묶음이다. 이 파일을 예컨대 C:\test\yahtex 폴더에 압축 해제하였다고 가정하겠다. 그러면 YAHTEXDOSBOX라는 폴더가 하나 생겨 있을 것이다.

2.2.1 DOSBOX 실행

DOSBOX 프로그램의 설치에 대해서는 더 말하지 않는다. 이를 실행한 후에 압축 해제한 폴더를 다음 명령으로 C: 드라이브로 마운트한다.

```
Z:\>mount c C:\test\yahtex\YAHTEXDOSBOX
```

그런 다음에 C:로 이동하여 AUTOEXEC.BAT를 실행한다.

```
Z:\>C:
C:\>AUTOEXEC.BAT
```

EmTeX을 위한 각종 변수 설정이 AUTOEXEC.BAT에 들어 있으므로 이를 반드시 실행해주어야 한다. 그 다음에 C:\MYTEX 폴더로 이동한다. 그림 2.1이 여기까지 진행한 것이다. (mount되는 폴더의 이름은 다를 수 있다.)

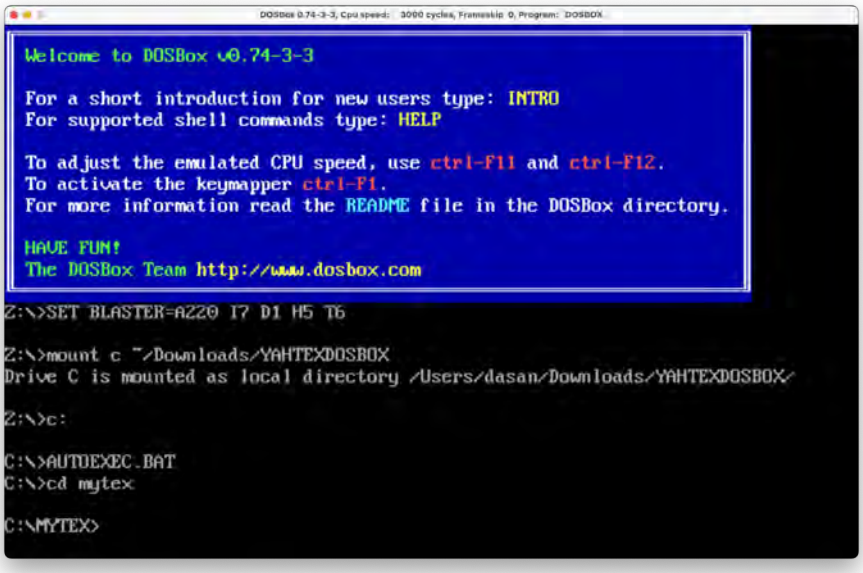

그림 2.1: DOSBOX 실행

2.2.2 hgh 실행과 편집, 컴파일

현재 이 폴더에는 MAN1.HTX라는 파일이 있을 것이다. (dir 명령을 내려보자.) 이 파일을 불러서 작업하는 '통합 작업 환경 배치파일' hgh를 이용하여 다음과 같이, 파일 이름만 확장자 없이 인자로 써준다.

```
C:\MYTEX>hgh man1
```

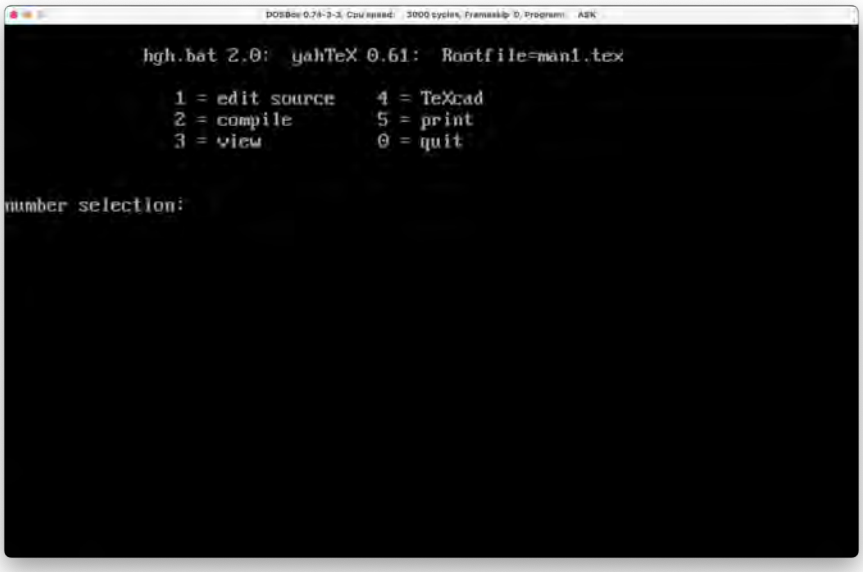

그림 2.2: hgh 실행

그림 2.2는 hgh가 실행된 상태이다. 여기서 할 수 있는 일은 그림을 보아서 알 수 있듯이 편집, 컴파일, 화면 미리보기이다. (print는 어차피 의미가 없고 texcad는 disable되어 있다.) 이 상태에서 빠져나가려면 0을 누른다.

1번을 누르면 MAN1.HTX가 uedit이라는 에디터로 열린다. 이 에디터의 메뉴는 <ESC>를 누르면 열리므로 편집하고 저장하는 등의 작업을 해볼 수 있다. uedit는 당시 굉장히 유명

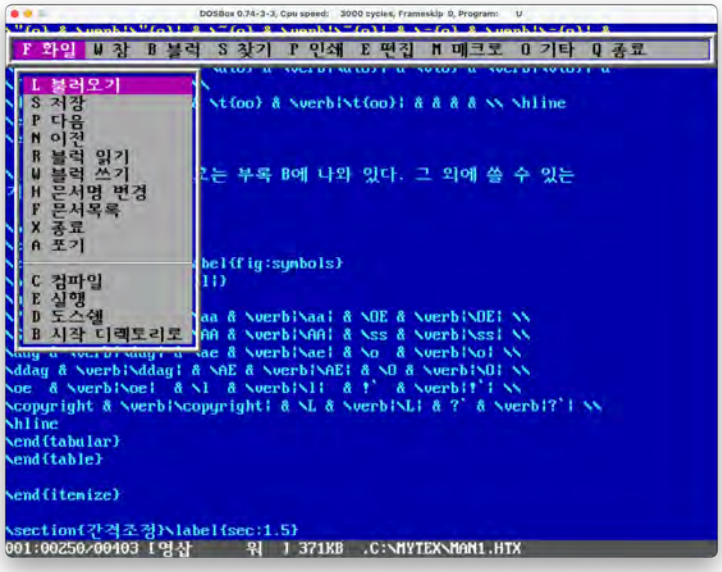

그림 2.3: 문서 편집

한 에디터였다. (에디터 내에 '컴파일'이라는 메뉴 항목이 있지만 TEX 컴파일을 의미하는 것이 아니므로 무시한다.) 그리고 앞으로 이 체험판 전체에 걸쳐 예제로 사용하게 될 MAN1 문서를 DOSBOX 내의 uedit에서 직접 입력하였다 그림 2.3. 한글 입력은 에디터가 담당했고 uedit에서는 Shift-Space로 한/영 전환을 할 수 있었다.

이 문서는 yahTeX에서 소스 없이 배포하는 '마누엘'이라는 문서(MANUEL.DVI)의 첫 장을 재현한 것이다. 왜 이름이 마누엘인지는 분명치 않지만 아마 "매뉴얼"이라고 하고 싶었던 것이 아니었을까? 그런데 이 문서는 내용이 꽤 충실해서 놀랍다. 요즘과 달리 1990년대에는 TEX 사용법을 안내하는 안내문서를 (비록 소스는 빼더라도) 반드시 포함시켜서 배포했던 것이다.

Alt-X를 눌러서 에디터를 빠져나오면 다시 hgh 화면으로 돌아온다. 2번을 눌러서 컴파일을 시도해보자. 운좋게 에러가 없으면 컴파일을 종료하고 첫 화면으로 돌아올 것이다.

이제 화면보기를 시도하자. 3번을 누르면 되는데, DVISCR라는 프로그램이 실행되면서 그냥 흰 화면만 보일 것이다. 이것은 불러들인 폰트의 해상도가 너무 높아서 그런 것이라서 - (마이너스) 키를 몇 번 쳐주면 적당한 크기로 보이게 된다. 화살표 키나 PgUp, PgDn 키로 이동할 수 있다. DOSBOX가 DOS 그래픽 화면을 처리하는 것이 아주 효율적이지 않고 불러들여야 하는 폰트가 너무 많으면 화면이 나타나는 데 시간이 좀 걸릴 수 있다는 사실을 알아두어야 한다. 참을성이 필요할지도 모른다.

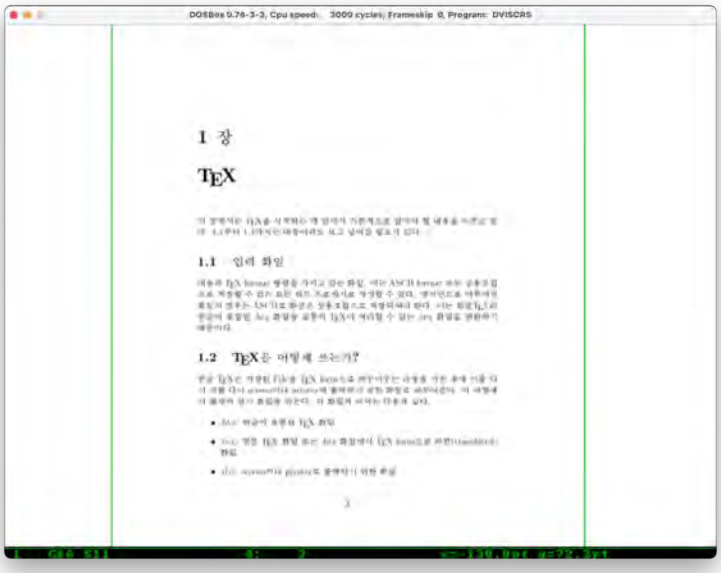

그림 2.4: DVISCR으로 미리보기

미리보기에서 Q 키를 누르면 빠져나간다.

파일 확장명이 HTX인 것은 특기할 필요가 있을 것이다. 한글이 포함된 소스는 이 확장명을 가졌는데 이것을 HTRAN.EXE이라는 유틸리티가 한글 부분을 번역해주었다. 한글을 일종의 매크로로 바꾸는 것이었다. 그러면 생성되는 파일이 TEX 확장명을 가지게 되고 이를 tex이 latex fmt를 불러서 컴파일하는 방식. 이것도 그나마 tex이 8비트를 처리할 수 있게 되었기 때문에 가능했다.

yahTeX 체험은 이것이 전부다. 에디터로 편집하고 컴파일하고 미리보기하였다. 당시에는 hgh의 메뉴로부터 프린트하는 것이 가능했을 것이다. LJ4 프린터 300dpi 해상도를 지원하였다.

여기서 만들어진 DVI 파일을 오늘날과 같이 PS나 PDF로 변환하는 것이 가능했을까? EmTeX에 DVIPS가 도입되었을 때 많은 사람들이 놀라워했지만 이 체험판에는 빠져 있다. 단지 약간의 수작업을 거쳐 현재의 dvips로 억지로 변환해본 결과가 Sample Documents에 포함되어 있다. 아무튼 PS, PDF는 이 당시 크게 중요한 문제도 아니었고 제대로 지원하지도 않았다.

2.3 Windows 3.1의 WinLaTeX

1992~3년경에 Windows 3.1이라는 놀라운 프로그램(이 때는 운영체제가 아니었다)이 출시되었다. 도스 사용자들이 군침만 흘리면서 바라보던 매킨토시의 "그림 인터페이스"가 마침내 DOS에도 도입된 것이다. 그리고 1993~4년 사이에 LG Software라는 회사가 Windows 3.1에 실행되는 WinLaTeX이라는 (당시로서는) 획기적인 프로그램을 판매하기 시작하였다.

사업 자체는 그다지 순조롭지 않았던 것 같다. 그걸 돈 주고 사느니 그냥 EmTeX 쓰고 만다는 것이었는지 별로 구매하는 사람이 많지 않았던 것 같고, 아무튼 결국 LG Software는 이 프로그램의 판매를 중지하고 1994년에 간략화한 버전(MathType을 제외)을 무료 다운로드 가능하도록 공개하는 것으로 결말이 지어졌다. 나중에 LaTeX 2ε 버전을 출시하겠다는 약속은 없었던 것이 되었다. 우리가 테스트해볼 것은 이 '공개' 버전이다.

특징을 요약한다.

(1) Windows 3.1에서 구동
(2) Windows의 한글 트루타입 폰트 활용
(3) TeX 3.141, LaTeX 2.09
(4) 한글은 전처리
(5) 상당량의 예제를 제공
(6) IDE. 단일 창 내에서 편집, 컴파일, 보기, 인쇄가 동작.
(7) DVIWIN이라는 dvi 뷰어 제공
(8) WMF 미디어 처리 가능

돌이켜보면 TeX은 매킨토시에서 활발하게 사용되었다. Bluesky의 Textures라는 '실시간 동기화 TeX 환경'은 1985년경(LaTeX도 없던 시절)에 시작되었고, OzTeX 역시 90년대가 되기 전에 확고한 자리를 잡았다. 이 시기 Windows 3.1 그림 인터페이스 환경이 도입되었을 때, 매킨토시의 OzTeX과 비슷한 작업환경을 만드는 것이 하나의 목표였을 것이지 싶다. WinLaTeX 역시 이러한 의도로 '통합 작업 환경' 프로그램을 만들지 않았을까? 그러나 Windows가 매킨토시를 따라잡으려면 한참 남았듯이, 이런 류의 Windows 프로그램도 생각만큼 편하지 않고 작업 효율의 향상도 뜻한 바와 같지 않은, 그런 상태였던 것이다.

WinLaTeX의 가장 큰 의의는 '트루타입 폰트의 활용'이라는 점이다. 물론 오늘날과 같은 의미에서의 트루타입의 활용은 아니고 트루타입에서 비트맵을 추출하여 화면에 디스플레이하고 인쇄에 사용하는 것이었지만 그 이전까지 소위 '폰트가 지원되지 않아서' 프린트를 할 수 없는 고통을 겪었던 EmTeX 사용자에게 반가운 소식이었다.

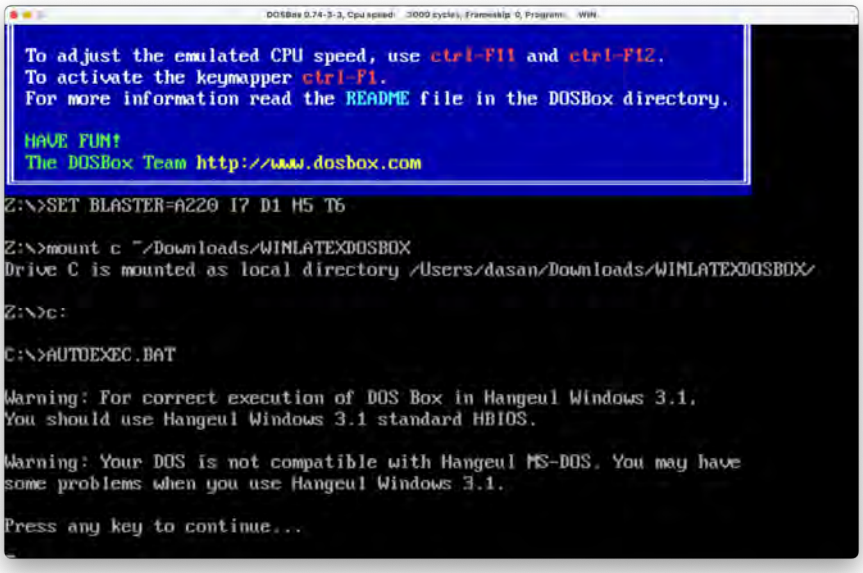

그림 2.5: DOSBOX로 Windows 3.1 실행

WinLaTeX의 한글 처리는 본질적으로 앞서 본 yahTeX의 그것과 큰 차이 없다. 전처리 변환기가 필요했고 그래서 한글이 포함된 LaTeX 파일의 확장명은 HLX, plainTeX의 확장명은 HTX였다.

2.3.1 DOSBOX에서 체험

1994-winlatex-win31_dosbox.zip를 다운로드받아서 예컨대 C:\test\winlatex에 풀어놓는다고 하자. 이름이 WINLATEXDOSBOX인 폴더가 생겨 있어야 한다.

DOSBOX를 실행하고 위의 폴더를 C:로 마운트한 다음 C:로 이동하여 AUTOEXEC.BAT를 실행하는 것까지는 같다. 그러면 그림 2.5와 같은 상태가 된다.

여기서 아무 키나 누르면 Windows 3.1이 구동된다. Windows 안에 WinLaTeX 그룹이 이미 설치되어 있다.

이 Windows 3.1의 한/영 전환은 소위 '유형 3' 키보드로서 Shift-space로 하게 되어 있다. '윈레이텍'이라는 프로그램이 TeX 작업을 위한 통합 환경이다. 파일을 새로 만들거나 불러오거나 하여 컴파일하고 미리보기(DVIWIN) 및 인쇄를 할 수 있었다. 체험판에서는 Postscript 프린터를 가상으로 설치하여 PS 파일로 인쇄하게 되어 있다.

불러오기 하여보면 C:\WINLATEX\SAMPLES\ATDOC\ 폴더에 man1.hlx를 발견할 수 있을 것이다. 이것은 yahTeX에서 만들었던 바로 그 파일인데, 확장명을 HLX로 수정하고 호환되지 않는 스타일 호출 부분을 살짝 수정한 것이다. 컴파일해보자. 사용자 감각으로는 yahTeX과 WinLaTeX이 거의 동일한 한글LaTeX 시스템임을 실감할 수 있을 것이다.

DVIWIN은 훌륭한 DVI 드라이버였다. 소문에 의하면 WinLaTeX을 출시하기 위해 LG Software사가 이 프로그램의 권리를 사들였다고 한다. WinLaTeX이 그렇게 결말이 났기 때문에 DVIWIN도 더이상 발전하기 못하고 여기서 사장되었다. 마우스의 가운데 버튼으로 확대경을 사용할 수 있었던 것이 흥미롭다.

그림 2.6: WinLaTeX in Windows 3.1

Sample Document는 '인쇄'로 얻은 `.prn` 파일을 `ps2pdf` 변환하여 얻었다. 300dpi 비트맵 글꼴로 인쇄된다.

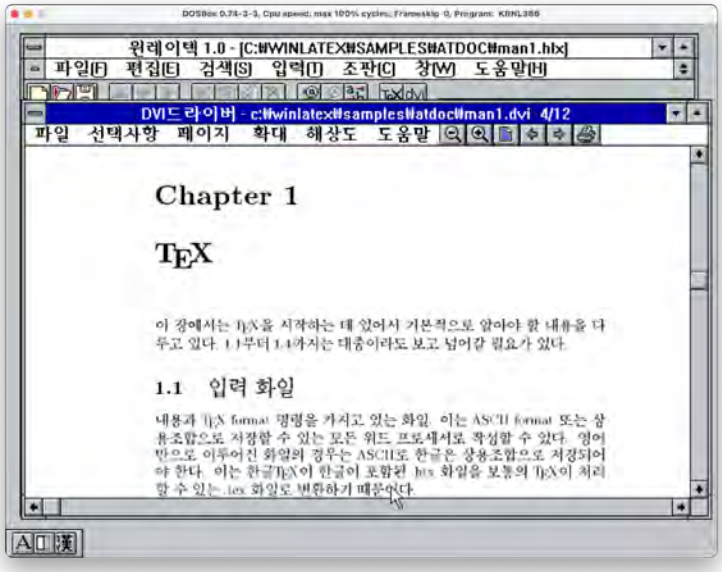

그림 2.7: DVIWIN으로 미리보기

WinLaTeX은 나름 훌륭한 면을 갖춘 통합 환경 프로그램이었으나 편집 기능이 조금 아쉬웠다. 그러나 `Ctrl-T`와 `Ctrl-D`로 컴파일하고 미리보기하는 과정은 상당히 매끄러웠다고 할 수 있다. DOS의 메모리 부족 때문에 죽기도 하고 중간에 컴파일 에러가 발생하면 수습할 수가 없던 등의 문제는 이 당시 TEX이 다 그랬으니……. 그리고 Windows 3.1 한글

입력의 고질적 문제로서 입력중인 글자가 에디터 창에 보이지 않는다는 점도 시선을 이탈시켜서 작업 효율을 낮추는 데 일조했다.

2.4 코멘트

8비트 TEX에서 한글을 처리하려는 노력이 처음 시작되었을 때는, 한글 코드의 각 바이트를 쪼개서 하나의 매크로로 묶어 전달하는 방식으로 해결하려 했던 듯하다. 이런 방식의 한글 처리를 우리는 '전처리 방식'이라고 부르고자 한다.

유감스러운 것은 yahTeX도 WinLaTeX도 전처리 프로그램을 exe 실행 파일로만 배포하였고 소스를 공개하지 않은 점이다. 그다지 어려워 보이지 않는 단순한 문자의 매크로 변환이었기 때문에 공개하지 않을 만큼의 신기한 기술도 아니었을텐데.

인쇄 품질로 말하자면 이 당시까지는 300dpi 프린터라면 최고급 사양이었다. yahTeX의 한글 서체는 불명이다. WinLaTeX은 Windows 3.1의 바탕체와 굴림체. 그러나 영문자와 한글 글자가 섞여 쓰이는 상황에서 그 높낮이와 크기를 적절하게 조절하지 못하여, 결과적으로 "어쨌든 한글이 인쇄는 되는" 정도에 그쳤다고 할 수 있겠다. 중요한 것은 '수식을 조판할 수 있는 워드 프로세서'라는 필요에 부응하는 것이었을 터이고, 적어도 그 점에서 기여한 바가 있다고 보겠다.

3

EUC-KR 한글의 시대: hLaTeXp

1995년부터 2005년경에 이르는 시기는 한글 LaTeX이 급격히 발전한 시기였다. 이와 더불어 한글텍사용자그룹(KTUG)의 활동도 활발해지고 한글TeX 사용자도 현저히 늘어나던 질적 양적 팽창의 생산적 시기였다고 할 수 있다. 이 시기를 대표하는 두 한글 매크로가 차재춘의 hLaTeXp, 은광희의 HLaTeX이다.

이 시기를 특징짓는 것은 '완성형 한글 코드'이다. Windows 운영체제가 주류로 되면서 완성형 한글은 자연스럽게 자리잡았다. LaTeX 역시 완성형 한글을 중심으로 활용성을 넓혀갔다. 그런 한편 이전 시기의 그다지 성공하지 못한 '상업화 시도'는 줄어들고 그 대신 TeX, LaTeX의 오픈소스적 가치가 더욱 중시되게 되었던 시기였다.

이 시기의 이야기를 둘로 나누어 이번 장에서는 주로 hLaTeXp와 관련된 것을, 그리고 다음 장에서 HLaTeX과 KTUG에서의 한글LaTeX의 발전에 대한 것을 다루겠다.

3.1 한TeX 1.5

워드 프로세서 훈글을 크게 성공시킨 한글과컴퓨터 사에서 한TeX이라는 프로그램을 출시하였다. 원래 의도한 이름은 '훈TeX'이 아니었을까 싶기도 하지만 이 글자를 찍을 방법이 없어서 '한TeX'으로 타협한 듯한 느낌을 준다. HTeX이라고 표기하기도 하였다.

이 프로그램도 처음에는 Windows 3.1에서 돌아가는 것이었다. 그러다가 1995년에 Windows 95가 나오고 나서, HTeX 1.5 버전을 새로 만들게 되고, 이 프로그램의 체험판을 『마이크로소프트웨어』라는 잡지의 부록으로 제공한 적이 있다. 우리가 살펴볼 것도 이 체험판이다.

3.1.1 한글LaTeX 시스템으로서의 한TeX

한TeX에 탑재되어 있는 것은 TeX 3.14159, LaTeX 2_ε이다. 여기에 hTeXn이라고 개발자들이 부른 "한글을 처리할 수 있는 TeX"의 LaTeX format인 hLaTeXn이 들어가 있었다. 이 시스템은 Windows 폰트를 기본 폰트로 사용하도록 조성된 것이었으며 TeX의 CM 폰트도 Windows용 트루타입으로 제공하고 있었다. 윈도우즈 기본 폰트 이외에 소위 엘렉스 폰트라고 하는 신명조, 중명조, 견명조 따위의 이름을 가진 트루타입도 함께 제공했다.

이 특징적인 트루타입 폰트 처리 부분을 표준 TeX의 METAFONT를 이용하는 것으로 교체한 것을 hLaTeXp라고 한다. 이에 대하여는 다음 절에서 더 살펴본다.

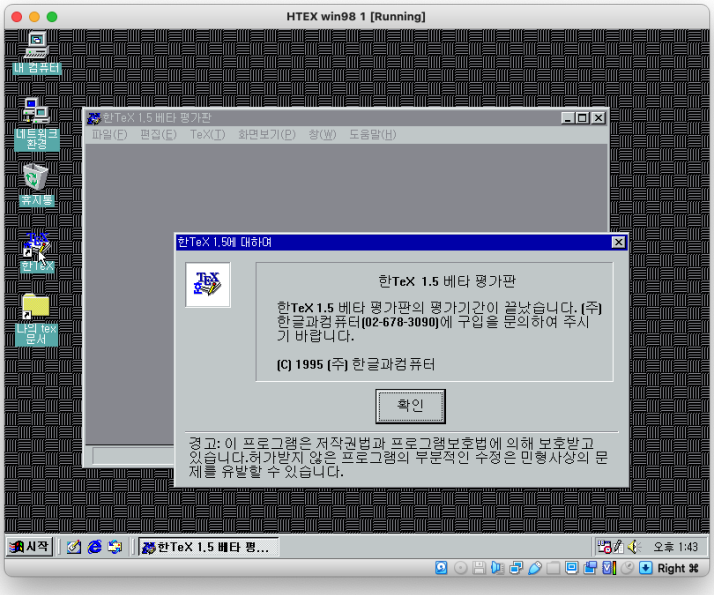

그림 3.1: Windows 98

한글은 완성형(KS C 5601)을 사용하였고 전처리기를 없앴다. 일반 텍스트로서 자연스럽게 한글을 입출력할 수 있었다. 이밖에 자동조사와 같은 중요한 한글화가 이루어졌다. hLaTeXn 또는 hLaTeXp로써 마침내 한글을 LaTeX 문서에서 자유롭게 쓸 수 있는 상태에 이르렀다고 해도 좋을 것이다.

Windows 95가 완성형 한글을 채택하였기 때문에 Windows 프로그램으로 제작된 HTEX으로서는 다른 선택의 여지가 별로 없었을 것이고, 이런저런 논란에도 완성형-조합형 논쟁은 사실상 완성형이 업계 표준의 자리를 굳혀가는 상황이었다. EUC-KR이라고도 불린 이 한글 코드는 나중에 Windows에 의해 CP949로 확장되게 되며, 매킨토시 측에서는 MacKorean이라 불리는 한글 코드페이지의 바탕이 되었다.

한TEX 역시 상업적으로 성공을 거두지는 못한 것으로 보인다. 한글과컴퓨터는 곧 이 사업을 접었다. 가장 결정적인 문제는 Windows 95에서 이 프로그램이 대단히 불안정하게 동작했다는 사실이다. 고질적인 메모리 누수 현상이 있었기 때문에 수 차례 컴파일을 시도하고 나면 블루스크린을 보게 되기가 일쑤였다. 프로그램 자체의 문제였는지 Windows 95 시스템의 문제였는지는 확실하지 않다. Windows 98에 설치하면 이런 장애는 줄었기 때문에 어쩌면 둘 다였는지도 모른다. (매킨토시용 한TEX은 상대적으로 안정적으로 돌았다는 말을 전해들었으나 확인은 해보지 못했다.)

사람들을 당황시켰던 것 중에 편집창이 익숙지 않은 것도 있었다. 커서 이동 방향이 행끝에서 다음 행 처음으로 이동하는 것이 아니라 아래위로만 움직였기 때문에 흔히 쓰던 에디터와 달라서 적응하기 쉽지 않았다. 편집기 자체 기능도 빈약해서 차라리 외부 에디터를 쓰는 것이 나았다. Notepad 정도의 기능만 제공했더라도 좋지 않았을까.

그림 처리에도 약간의 문제가 있어서 wmf 그림의 정확한 크기가 인쇄에 반영되지 않는 경우도 있었다고 한다. 특히 고급 포스트스크립트 프린터에서 심했는데, 그 결과 화면으로는 제대로 보이는 페이지가 인쇄하면 이상해지기도 하고.

이런 여러 문제를 가진 프로그램이었으나 한글 LATEX 시스템 자체만을 놓고 보자면 대단히 완성도 높은 축에 들었고 TEX 대중화에 어느 정도 공헌한 바도 있어서 역사적 의미가 작다고 하기는 어렵다.

3.1.2 VirtualBox를 통한 체험

1995-HTEX-win98_vbox.ova (129M) 파일은 ova (가상시스템) 파일이다. 한글 Windows 98 SE에 HTEX을 설치한 상태로 제공한다. Windows 98에 설치한 이유는 Windows 95에서는 시스템이 너무 자주 다운되어서 도저히 테스트할 수 없는 상태라고 판단했기 때문이다. 이 ova는 VirtualBox에서 가져올 수 있다. (현재 버전의 VMware Player로는 Windows 98이 잘 동작하지 않았다.) 새로운 가상 시스템으로 가져온 다음 그대로 실행하면 된다. Windows 로그온 비밀번호는 따로 없기 때문에 그대로 엔터를 쳐서 진행한다.

바탕화면의 HTeX을 더블클릭하여 실행하거나, "나의 TeX 문서" 폴더 안의 예제 문서 02-HTEX.tex을 더블클릭해도 한TEX이 열린다. 재미있게도 문서를 더블클릭했을 때 파일을 불러오지는 않지만.

여기 들어 있는 예제 02-HTEX.tex은 yahTeX에서 작성하고 WinLaTeX에서도 테스트했던 그 문서이다. 내용은 똑같고 처음 한두 줄만 고쳐서 테스트용으로 쓰고 있다. F5를 눌러서 컴파일을 시도해보자.

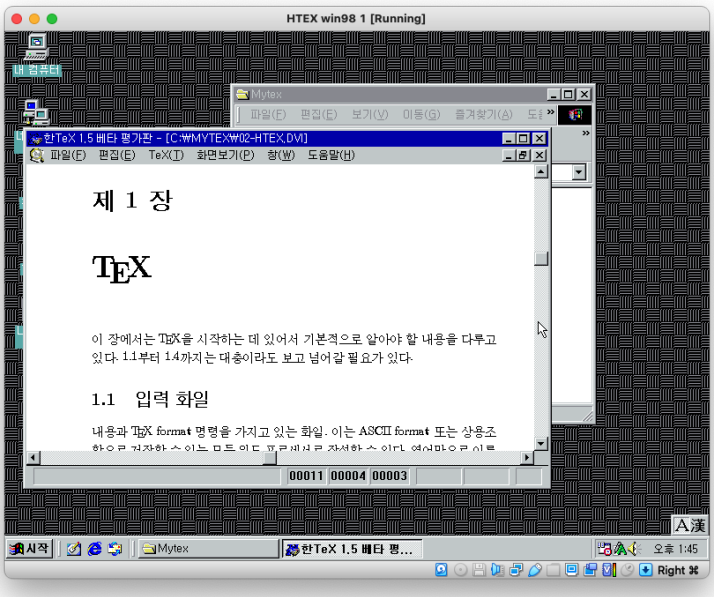

그림 3.2: HTEX 화면보기

HTEX에 오면 오늘날 익숙한 LATEX 문서에 가까운 형식으로 문서를 작성할 수 있다. 샘플 문서를 C:\HTEX15\SAMPLES 폴더에서 찾을 수 있으므로 이것저것 시험해보기 바란다. 가상 Postscript 프린터를 설치해두었기 때문에 인쇄하여 PS 형식인 .prn 파일을 얻을 수 있다.

3.2 hLaTeXp: OzTeX에 설치

hLaTeXn과 달리 hLaTeXp는 특정 운영체제 의존적이지 않은, TeX이 설치되어 운영되는 곳 어디라도 매크로 패키지로서 설치하여 한글 문서를 작성할 수 있다는 중요한 특징을 가졌다. 운영체제나 하드웨어와 상관없이 TeX 시스템 위에 설치할 수 있다는 이 특징 때문에, 그간 선진적인 환경의 TeX을 가졌으면서도 한글 사용에 제약이 있던 매킨토시 사용자들도 마침내 한글 패키지를 설치할 수 있게 되었다. 물론 Windows의 MiKTeX이나 fpTeX, Linux의 teTeX에도 한글 환경을 구축하는 것이 가능해졌다. 이것은 매우 중요한 일보전진이었다.

그런데 안타깝게도 hLaTeXp는 METAFONT 시스템을 이용하는 매크로 패키지이면서도 막상 METAFONT 폰트 소스 자체를 공개하지 않았다. 그 까닭은 짐작건대 이 폰트들을 연구개발용으로 제공하였을 엘렉스 컴퓨터 측에서 계약상 공개를 제한하였기 때문이 아니었을까 한다. 그 대신, 미리 생성된 300dpi, 600dpi PK 비트맵 폰트를 별도로 제공하는 방식을 썼는데 그 결과 문서의 품위도 품위려니와 높은 해상도 문서를 작성할 수 없게 되는 등 제약이 발생하고 말았다. 당시의 컴퓨팅 환경을 고려하고 TeX 문서 작성의 목적이 프린터 출력이었음을 생각하면 이 정도로 만족하면서 쓰는 것이 불가능하지는 않았겠으나, 날이 갈수록 이 제한은 답답하게 여겨졌고, 몇년 지나지 않아 pdf 폰트 임베딩이나 텍스트 추출 검색 등이 문제가 되기 시작하면서 차차 사용자의 외면을 받게 되는 것을 피할 수 없었다.

3.2.1 매킨토시와 OzTeX

1996-OzTeX-Mac755_SheepShaver.exe.zip (384M) 파일은 Windows에서 실행할 수 있는 자동풀림압축파일이다. 실행하면 압축을 풀고 SheepShaver라는 파워맥 에뮬레이터 위에 설치된 매킨토시 한글시스템 7.5.5 위에 OzTeX과 hLaTeXp가 설치된 상태를 보여준다. '도큐멘트' 폴더에 있는 untitled-1.tex은 이 가상 기계를 가지고 노는 방법을 자세히 적은 것으로 한번 읽어보면 좋다.

호스트 컴퓨터와의 파일 교환이 매우 쉽기 때문에 별도로 예제 파일을 넣어둔 상태로 제공하지 않는다. SampleDocument에서 03-hltxp_OZ.tex을 가져와서 직접 컴파일해보기 바란다.

이 체험에서 중요한 것은 한글LaTeX 자체가 아니고 매킨토시에서 얼마나 편리하게 TeX 작업이 이루어졌는가 하는 것이다. 컴파일은 에디터 내에서의 단축키나 "끌어다놓기"로 이루어진다. 오늘날과 비교하면 바로 DVI 화면에서 해당하는 에디터의 행으로 직접 간다든가 하는 (당연해보이지만 놀라운) 기능은 아직 없지만 미려한 그림 운영체제와 직관적인 조작으로 작업자를 편하게 해주었다. 거기에 더하여 매우 발전한 그림 처리 기술 (GraphicsConverter로 대표되는)과 이를 바탕으로 하여 고품위의 삽입 그림을 가진 결과물을 만들어낼 수 있었던 것도 언급할 수 있겠다. 그리고 거기에 더하여 이제 한글도 쓸 수 있게 되었던 것이다. (다만 그 유명한 Alpha 에디터에 한글을 사용할 수 없었던 것은 OzTeX 활용성을 조금 낮추는 결과를 가져왔다고 보지만 이외에도 TeX 작업에 쓸 좋은 에디터가 많았다.)

유감인 것은 이렇게 훌륭한 환경에서 만들어내는 한글 문서의 폰트 품위가 낮다는 사실이었다. 이것은 hLaTeXp가 폰트를 그런 식으로 공개하였던 바로 그 사실 때문이므로 안타깝다고밖에 말할 수 없다.

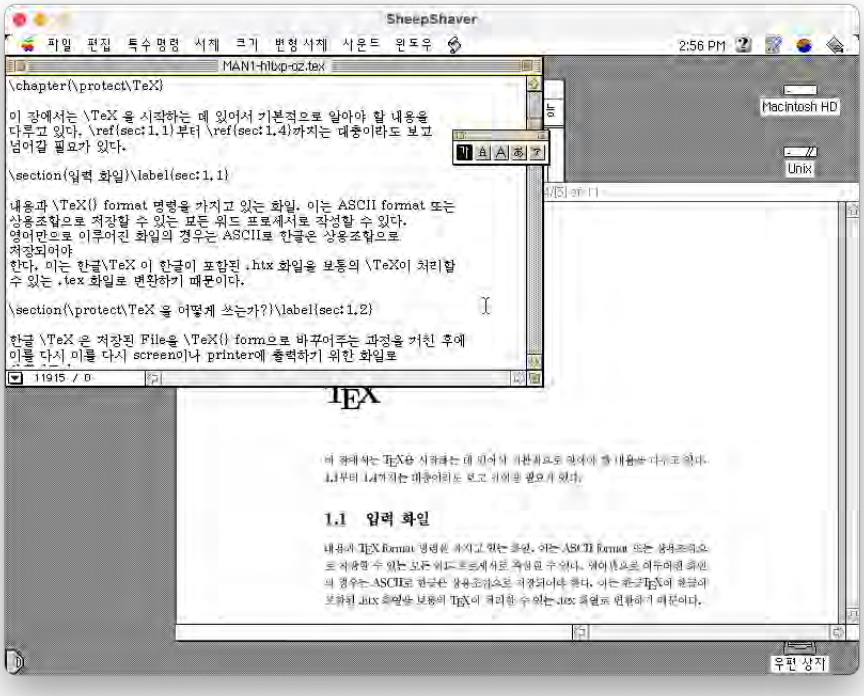

그림 3.3: Tex-Edit과 OzTEX 실행화면

어차피 OzTEX은 TEX 시스템이고 hLaTeXp나 HLaTeX이 모두 매크로 패키지이므로 포스트스크립트 글꼴을 쓸 수 있었던 HLaTeX을 설치하면 폰트 문제의 사정이 조금 나아지기는 하지만 근본적으로 XeTeX의 시대에 와야—여기서 XeTeX이란 것은 오직 매킨토시에서만 동작하던 초창기 XeTeX을 말한다. 매킨토시의 폰트를 TEX에서 써보자는 것이 이 새로운 엔진의 시발점이었던 것이다—해결될 문제였다.

이 당시 매킨토시 시스템 7은 MacKorean이라는 한글 코드 체계로 운용되었다. 이것은 근본적으로 EUC-KR과 같은 것이어서 hLaTeXp와 무리없이 결합할 수 있었다. 매킨토시는 MacOS X이 나오고서야 완전히 유니코드 베이스로 이행한다.

3.3 hLaTeXp

지금 최종적으로 확인할 수 있는 hLaTeXp 공개 버전은 1998년에 나온 것이 마지막이다. 다음 장에서 설명할 Linux TEX Live에서도 hLaTeXp를 설치할 수 있고, 동일한 결과를 얻는다.

다음 장에서 소개하는 체험용 가상 시스템(Linux)에 hLaTeXp도 설치되어 있다. 그런데 HLaTeX의 패키지 이름도 `hangul.sty`이고 hLaTeXp에도 `hangul.sty`이 있어서 이 둘을 동시에 설치하기 위하여 hLaTeXp의 `hangul.sty`를 `hangulp.sty`로 바꾸어두었다. Xubuntu 가상 기계에는 hLaTeXp 테스트 파일이 빠져 있으므로 이를 테스트하려면 먼저 `11-HLaTeX.tex`을 `hlatexptest.tex`으로 복사한 다음 preamble의 `\usepackage{hangul}`을

```
\usepackage{hangulp}
```

로 바꾸어 써넣든가 아니면 아예 한글 패키지 로드하는 행을 없애버리고

```
hlatex hlatexptest.tex
```

으로 컴파일하여 dvips할 수 있다. xdvi로도 잘 볼 수 있을 것이다. hangulp 패키지가 있고 없고의 차이는 행간격, 명칭, 날짜, 장절 이름 등의 한글화가 적용되느냐 되지 않느냐이다.

그림 3.4: Linux 가상기계 내에서 hLaTeXp, xdvi

이와 같이 hLaTeXp는 latex 명령 대신 hlatex으로 컴파일하여야 한다. 이것이 의미하는 바는 hlatex.fmt라는 포맷을 로드하여 컴파일하라는 뜻이다. 그 결과 아무런 패키지를 로드하지 않아도 한글이 처리되는 것처럼 보인다.

재미삼아

```
pdfhlatex hlatexptest.tex
```

도 실행되게 해두었다. pdftex으로 hlatex.fmt를 조성한 것인데 이것은 체험판을 만들면서 필자가 추가한 포맷이다. PDF를 바로 얻을 수 있다. 그러나 어느 경우든 결국 한글 폰트는 pk 비트맵일 따름이다. pdfhlatex을 실행할 효용이 별로 없다.

4

HLaTeX, 한글 LaTeX의 발전

4.1 HLaTeX

한글LaTeX의 발전사에서 HLaTeX은 매우 중요한 위치를 차지하고 있다. 대체로 1998년 무렵 이후에는 이것이 한글 쓰기의 "사실상 표준" 취급을 받았으며, KTUG에서 이 패키지를 바탕으로 당시 발전하고 있던 새로운 환경에 적응하기 위하여 다양한 실험을 행하였다. 그 성과를 토대로 하여 만들어진 것이 오늘날의 ko.TeX이다.

HLaTeX으로써 KTUG에서 이루어진 다양한 실험과 발전에 대해서는 별도의 글이 필요할 것이다. 여기서는 원본 HLaTeX의 주요 특징을 간추리고 지나가려 한다.

(1) `fmt`을 형성하여 한글을 처리하던 hLaTeXp와 달리 완전한 매크로 패키지였다. 그래서 한글이 문서에서 나타나게 하려면

```
\usepackage{hangul}
```

또는

```
\usepackage{hfont}
```

라는 선언이 반드시 필요하였다.

(2) 0.98까지는 METAFONT, 0.99부터는 `pfb` type 1 폰트가 기본으로 제공되었다. 은-시리즈(uhc) 글꼴이라 불린 이 type 1 폰트로부터 '은 글꼴'이라는 트루타입이 만들어졌고, 이 과정은 특히 PDF를 LaTeX의 최종 출력물로 하는 새로운 상황에서 그 유용성이 입증되었다.

(3) 1.0 이후 버전에서는 완성형 한글의 한계를 벗어나서 UHC 글자를 전부 표현할 수 있도록 하려고 당시 개발 중이던 Omega 시스템에 주목하였다. 이 Omega는 개발이 중단되고 현재의 TeX 시스템에는 흔적만 남아 있으며 유니코드 TeX은 XeTeX, LuaTeX 등으로 발전하게 되지만 한글 표현 문제에 대한 치열한 고민이 남아 있으므로 귀중한 유산이다.

(4) 버전명에서 볼 수 있듯이 오랜 시간에 걸쳐 패키지가 관리되고 유지·보수되었다.

(5) LaTeX 사용자가 늘어나던 시기와 맞물려서 많은 사람에게 익숙하게 받아들여졌다.

4.2 OzTeX에서 HLaTeX 0.98

HLaTeX은 0.98까지 wansung이라는 이름의 폰트를 제공하였는데, 이것은 METAFONT 소스를 포함하는 것이었다. wansung 폰트는 문체부 글꼴을 바탕으로 작성된 것으로 보이고 실제 문체부 트루타입과 함께 제공하였다. 0.99 이후로 이 폰트는 더이상 HLaTeX에 포함되지 않고 UHC type1 글꼴로 대체되었다. (UHC 폰트는 0.98에 이미 들어 있었으며 새로운 폰트를 적용할 준비가 갖추어져 있었다. 이 글에서는 단지 METAFONT에 대해서만 언급한다.)

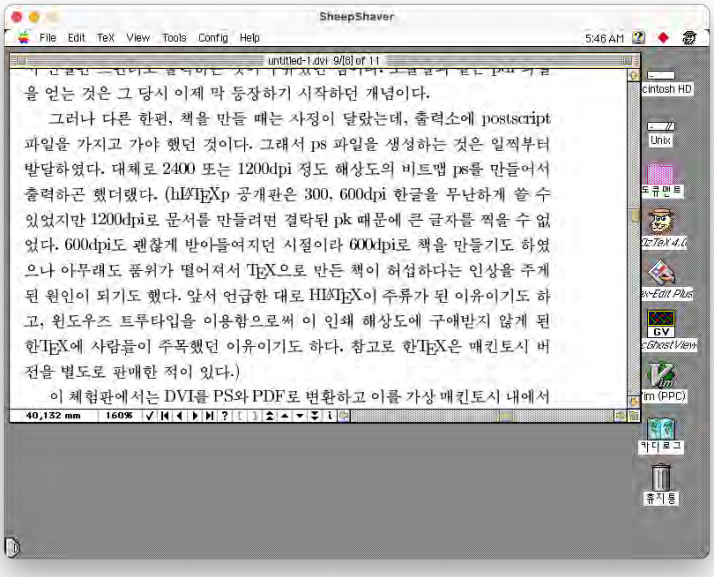

그림 4.1: HLaTeX 0.98의 wansung 폰트

일반 사용자에게 아마 METAFONT가 PK 비트맵을 생성하는 것을 보는 것은 매우 색다른 경험이었을 것이다. 왜냐하면 그 이전까지 모든 한글 LaTeX 시스템이 미리 조성된 PK 글꼴만을 제공하였기 때문이다. 정해진 거 말고 임의의 사이즈로 폰트를 식자할 수 있다는 것만 해도 상당한 만족감을 주었던 것이다.

OzTeX 4.0은 별도의 추가 설치 없이 type1이나 트루타입을 즉시 사용할 수 없었기 때문에 METAFONT 글꼴의 존재가 아주 절실했다. HLaTeX의 매크로와 wansung 폰트 부분만을 체험판에 설치하여, 이 '새로운 경험'이 어떤 것이었는지 짐작해보게 하였다.

4.3 PDF Driver와 HLaTeX

당시 LaTeX 사용자가 직면한 문제 중에 고품위의 PDF 제작이라는 것이 있었다. 인쇄 출판업계도 서서히 Postscript에서 PDF로 출력 포맷이 바뀌어가고 있었으며, 새로운 온라인 문서 표준으로 PDF가 받아들여지기 시작하던 시절이었다. 그때까지의 한글LaTeX들로는 (이제까지 보아온 대로) 비트맵 폰트만을 사용할 수 있었기 때문에 해상도가 300 내지 600dpi를 넘어설 수도 없었고 텍스트 변환이나 검색 추출이 되지 못하는 한계가 있었다. 우리는 윤곽선 글꼴을 내장한 고품위의, 한글 텍스트의 검색과 추출이 되는 PDF를 LaTeX으로 제작하고 싶었던 것이다.

HLATEX에서 그 가능성을 발견했다. 그리고 당시 각광받고 있던 pdfTEX이라는 새로운 엔진으로도 무리없이 PDF 출력물을 얻을 수 있었다. 다른 한편, dvipdfm과 이를 조진환이 발전시킨 dvipdfmx는 트루타입 폰트와 HLATEX을 연결해서 쓰는 방법으로 원하는 결과를 얻을 수 있음을 보여주었다. 이것은 당시로서 굉장한 발전이어서 새로운 PDF 제작 방법으로 각광을 받았다. dvipdfmx는 주로 일본에서 pTEX과 함께 쓰이다가 현재는 XETEX의 유일한 dvi (xdv) 드라이버로 그 명성을 유지하고 있다.

그 결과, 한동안 LATEX의 출력물을 얻는 방법이 다음과 같이 여러 가지가 공존하게 되었다.

latex → dvips → ps2pdf 전통적 방법으로 얻어진 DVI 파일을 dvips로 처리하고 이로부터 얻어진 PS 파일을 ghostscript로 PDF 변환하는 방법이었다. 이 방법은 번거로워서 점점 사용층이 줄어갔는데, EPS와 PSTricks라는 그림언어를 처리하기 위해서는 이 방법 밖에 없어서 꼭 써야할 때가 있었다.

latex → dvipdfm(x) 전통적 방법으로 얻어진 DVI 파일을 dvikpdfm 또는 dvipdfmx로 변환하는 방법이었다. 특히 한글 트루타입을 내장한 PDF를 만들려면 dvipdfmx를 써야 했다.

pdflatex pdfTEX 엔진으로 직접 PDF를 얻는 방법이었다.

어떤 경로로 출력물을 얻을 것인가에 따라 드라이버에 민감한 패키지(예를 들면 graphicx)는 옵션을 잘 설정해야 하는 문제가 없지는 않았고, 각 드라이버에 따라 EPS 그림을 처리할 수 있느냐 없느냐의 문제가 있기는 했다. "pdfTEX은 EPS 그림을 처리하지 못합니다"는 답변을 수없이 반복하던 시절이 있었다. 이제 EPS를 아무도 찾지 않고 PNG, JPG가 사실상 표준처럼 받아들여지는 세월이 되고 보니, 금석지감이 없다 하지 못하겠다.

그리고 이렇게 선택 가능성이 다양해지고 이에 대응할 수 있었던 것은 HLATEX이 폰트를 제공했기 때문이다.

4.4 체험용 TEX Live 가상 시스템

한편 그 이후 한글 LATEX의 발전을 체험하기 위해 준비한 리눅스 가상 기계는 다음과 같이 구성하였다.

- Xubuntu 16.04
- TEX Live 2008 Full
- kɷ.TEX 개발 이전까지의 한글 패키지들

이 파일은 `1998-prekotex-xubuntu16.04_vbox.ova` (4.5G)에서 다운로드할 수 있으나 파일 크기가 매우 크다는 점에 주의하라. ova 파일이므로 VirtualBox나 VMware player (fusion)에서 import하여 실행할 수 있다. 기본 사용자는 `texer`, sudo 패스워드는 0000이다.

Xubuntu 16.04를 선택한 이유는 그 이후 버전에서는 라이브러리 충돌 때문에 TEX Live의 xpdf를 실행할 수 없음을 확인하였기 때문이다. 이 리눅스 자체가 제공하는 texlive 패키지는 texlive 2015인데, 이를 무시하고 TEX Live 2008 버전을 full로 별도 설치하였다. 대체로 말해서 TEX Live 2021 이전 버전들은 거의 유사하게 동작하는 것으로 볼 수 있지만 패키지 볼륨에 큰 차이가 있고 일부 예전 패키지와 미묘하게 걸맞지 않는 부분이 있는 것 같다.

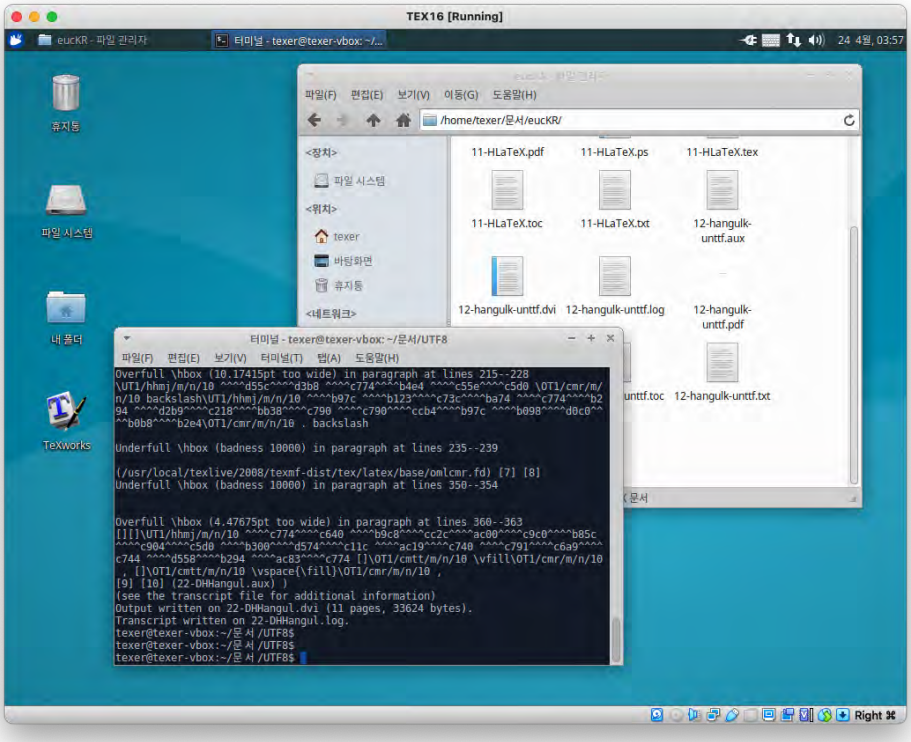

그림 4.2: Xubuntu 16.04와 TEX Live 2008

여기에 설치된 한글 패키지들은 hLaTeXp, HLaTeX를 기본으로 하여 훗날 ko.TEX 개발의 바탕이 된 실험적이고 과도기적인 패키지들이 포함되어 있다. 현재의 TEX 시스템으로는 돌려볼 수 없는 Lambda (Lamed) 패키지가 들어 있다.

이 가상 시스템 내에서 한/영 전환은 `Ctrl-Space`이다.

4.4.1 HLaTeX 컴파일 시도

테스트 가상 기계를 실행하고 `~/Documents` 폴더로 이동하면 두 개의 하위 폴더가 마련되어 있다. 이 중에서 HLaTeX 테스트 파일은 `eucKR` 폴더에 들어 있다.

이 기계의 시스템 로케일은 `ko_KR.UTF-8`인데, 지금 편집하거나 컴파일하려는 파일은 인코딩이 EUC-KR이다. 터미널 에디터로 한글 부분을 편집하는 게 귀찮을 듯해서 TEXworks를 깔아두었다. 다음 magic comment로 EUC-KR 문서도 무리없이 편집할 수 있을 것이다.

```
%!TEX encoding = EUC-KR
```

나중에 UTF-8 문서도 다루어야 하기 때문에 시스템 로케일을 바꾸는 방법을 쓰지 않았다.

HLaTeX 기본 예제 문서는 `11-HLaTeX.tex`인데 yahTeX에서부터 가져온 그 파일이다.

pdfTEX 컴파일 TEXworks에서 'pdfLaTeX'으로 컴파일해보자. 에러없이 진행되고 SyncTeX에 의한 마주 찾기도 잘 될 것이다.

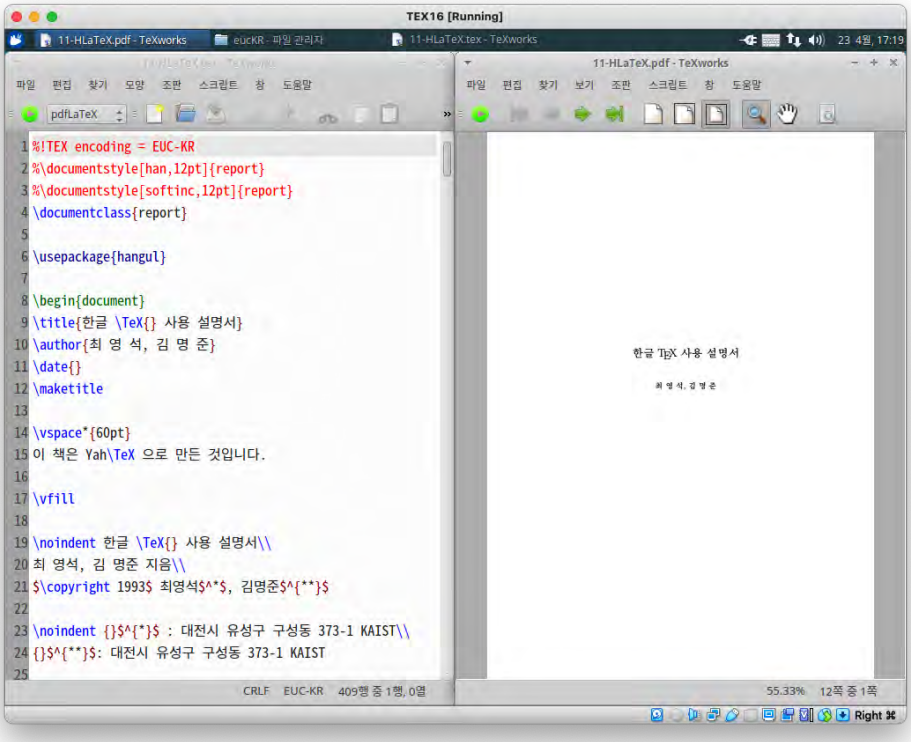

그림 4.3: TEXworks로 EUC-KR 문서 편집

dvips 변환 이번에는 터미널을 열고

```
latex 11-HLaTeX.tex
dvips 11-HLaTeX.dvi
```

명령을 차례로 내려보자. ps2pdf 스크립트로 PDF 변환할 수 있다.

dvipdfm 변환 이 샘플 파일은 dvipdfm으로 변환하면 된다.

```
latex 11-HLaTeX.tex
dvipdfm 11-HLaTeX.dvi
```

이렇게 얻어진 PDF는 uhc type1 폰트를 내장하고 있다. 검색 추출에까지 이르지는 않지만 이전에 상상할 수 없던 결과를 얻을 수 있는 것이다.

흥미가 있다면 PSTricks 그림, EPS 그림 등이 dvips를 이용하여야만 동작한다는 것을 확인해보는 것도 좋을 것이다.

4.4.2 unttf, hangul-k

박원규 등의 '은 글꼴'은 은광희의 HLATEX에 포함되어 있던 type1 글꼴 uhc를 트루타입으로 변환한 것이었다. 이 글꼴의 등장이 중요한 것은, 이를 통하여 "검색 추출 가능하고 내부 참조가 되는 PDF" 제작에 한 걸음 다가갔다는 사실이다.

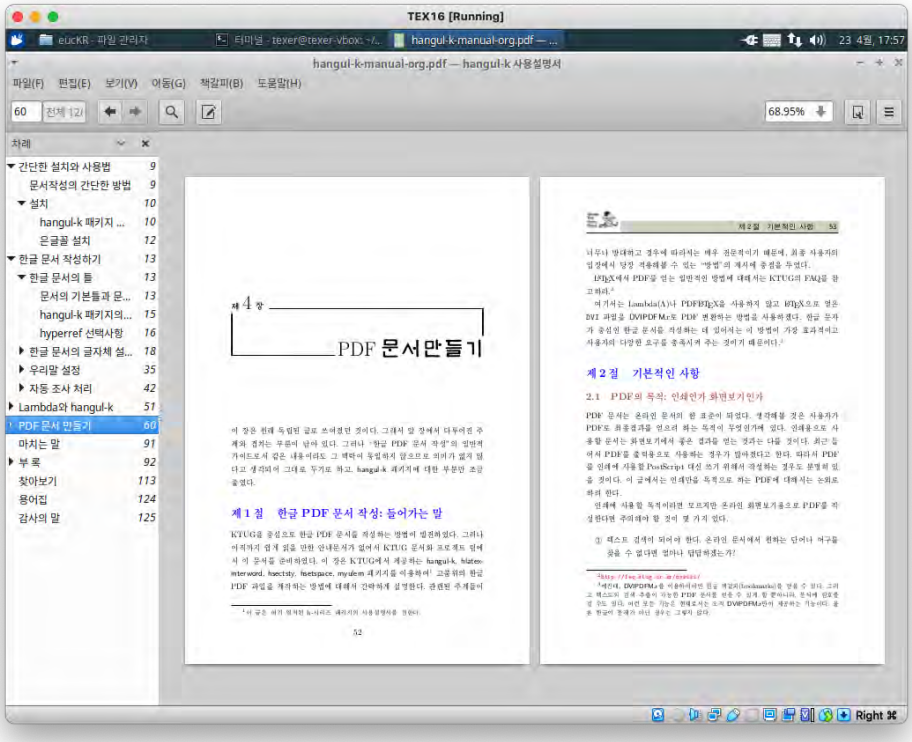

그림 4.4: hangul-k 매뉴얼

테스트 파일은 `12-hangulk-unttf.tex`이다. 은 글꼴 트루타입을 사용하도록 설정한 문서이다. pdfLaTeX으로도 소스에 지정된 map 파일을 이용하여 컴파일이 잘 이루어질 것이다. 다음 코드에 주목하라.

```
\usepackage{ifpdf}
\ifpdf
    \pdfmapfile{=unttf-pdftex-hlatex.map}
\fi
```

그러나 이렇게 만들어진 PDF를 `pdftotext` 유틸리티로 텍스트 변환하면 한글이 제대로 나타나지 않는다.

여기서 당시 `dvipdfmx`가 주목받은 이유를 확인하자. 다음 명령을 터미널에서 내려서 결과를 확인해보라.

```
latex 12-hangulk-unttf.tex
dvipdfmx 12-hangulk-unttf.dvi
```

결과를 확인하기 위해서 `pstotext` 유틸리티로 생성된 `.txt` 파일을 텍스트 에디터에서 읽어보면 된다. 심지어 EUC-KR 소스에서 얻은 PDF인데도 txt 변환한 결과는 UTF-8이다.

`hangul-k` 스타일은 `hangul.sty`를 바탕으로 수정을 가한 KTUG에서 만들어진 패키지이다. `hyperref`에 의한 하이퍼링크와 북마크, 자동조사 등 당시 현안이었던 몇 가지 문제를 보완하려 했던 것으로 기억한다. 실제 사용된 것은 얼마 되지 않지만 '도은이아빠'가

쓴 매뉴얼이 제법 유명했고 ko.TEX-euc 패키지의 원형이 되었던 것이다. 매뉴얼을 읽으려면 `texdoc hangul-k` 명령을 내려보라.

그림 4.4와 같이 깔끔한 한글 bookmark가 온전하게 나타나는 것이 이 당시에는 자랑할 만한 일이었다.

5

Unicode UTF-8 한글 LaTeX

일찍부터 완성형 한글의 2350자 제한은 이런저런 소동의 원인이 되었다. 조합형주의자들이 이때까지도 포기를 못하고 완성형 한글의 문제점을 지적하는 일이 심심찮게 계속되었는데, 예컨대 당시 어느 방송사의 인기 드라마 제목인 "똠방각하"의 '똠'이 완성형 한글 중에 들어 있지 않아서 표기할 수가 없다든가, 옛한글이 완전히 도외시된다든가, '믠'이나 '띡'이 없다든가, 소란이 끊이지 않았다. 역설적이지만 워드 프로세서 흔글이 성공한 이면에 이런 제한적인 한글 표현을 넘어서서 문서 작성이 가능하다는 점이 있었다는 주장도 있으니 재미있다면 재미있는 일이다.

완성형 한글 코드를 기반으로 하고 있는 한글LaTeX으로는 결국 이러한 한계를 극복할 수 없었다. 완성형 밖의 한글과 옛한글의 LaTeX 구현은 당시 LaTeX 사용자의 꿈이었고, 유니코드 TeX 엔진의 시대가 되어 최종적으로 해결된다. 그 과도기의 시도들에 대해 알아보는 것이 이번 장의 목표이다.

5.1 dhucs

KTUG에서 이루어지던 ucs 패키지와 inputenc 패키지에 대한 실험을 토대로 마침내 UTF-8 입력에 대한 HLaTeX의 확장이 이루어졌다. 2004년 전후한 일이었다. 김도현이 UTF-8 한글 처리 부분의 핵심 코드를 짜고, 김강수가 hangul-k에서 이미 정비가 이루어진 매크로를 적용하여 새로운 패키지가 하나 탄생했으니 이것이 hangul-ucs (dhucs)라는 것이었다. 이것은 그 후 ko.TeX-utf의 근간이 되었고, 실제로 아직도 `dhucs.sty`라는 이름으로 ko.TeX-utf 패키지 속에 남아 있다.

이 체험판에 포함한 것은 ko.TeX-utf의 dhucs가 아니고 그 이전에 처음 작성되었던, 최초로 유니코드 *UTF-8* 입력 한글을 처리하는 한글 *LaTeX* 패키지였던 `dhucs.sty`이다. 필자는 이것을 KTUG 최대 성과 중의 하나라고 생각한다.

5.1.1 테스트 파일

UTF8 폴더의 `20-dhucs.tex` 파일이 dhucs를 테스트하기 위한 파일이다. 당연히 UTF-8 인코딩으로 입력되었으므로 편집에는 문제가 없을 것이다. 이 파일은 latex 컴파일하고 dvipdfmx로 변환하여 검색 추출 가능한 PDF를 만들 수 있다. 다만 다음과 같이 `map` 옵션을 주어야 한다.

```
latex 20-dhucs.tex
```

```
dvipdfmx -f cid-unttf-dhucs.map 20-dhucs.dvi
```

dvipdmfx 실행시에 특정 map 파일을 강제 지정하게 하는 이 방식은 `dvipdfmx.cfg`를 수정하여 자동 처리되게 할 수 있었으나, 테스트 가상 시스템에서 이 부분을 제외해두었기 때문.

체험판에 수록된 dhucs는 (훗날 ko.TEX의 그것과 달리) 은 글꼴 트루타입을 기본으로 하고 dvipdfmx PDF 변환하는 것을 전제로 하고 있다. 그래도 pdflatex으로 컴파일하면 오류가 발생하지는 않도록 하는 코드를 포함하고 있다.

한편 유니코드 UTF-8 입력된 소스에 대하여, pdfLaTeX으로 컴파일하는 경우의 텍스트 검색 추출 문제는 나중에 다음과 같이 하여 해결하게 된다. 참고로 적어둔다.

```
\usepackage{ifpdf}
\ifpdf
    \input glyphtounicode
    \pdfgentounicode=1
\fi
```

5.2 Omega로의 횡로(橫路)

Omega란 John Plaice와 Yannis Haralambous가 개발한 것으로 16비트 문자 체계를 수용하여 다시 쓴 TEX이었다. 전 세계 모든 문자를 TEX으로 식자할 가능성이 있다고 생각되어서 개발 초기부터 많은 관심을 받았으나 2004년에 John Plaice가 새로 시작하겠다고 선언한 이후 사실상 개발이 중단되었다. Omega에 ε-TEX 엔진 매크로를 적용한 시스템을 Aleph라고 한다. Omega의 LaTeX 포맷을 Lambda라고 부르고 Aleph의 LaTeX 포맷을 Lamed라고 하는데 —Lamed는 L에 해당하는 히브리 자모 이름에서 온 것으로 "레임드"라고 읽으면 안 된다. "라메드."—, 일부 아랍어권에서 Omega (Aleph)에 기초한 조판 솔루션을 발전시켜 실용적으로 쓰는 단계까지 왔었기 때문에 TEX Live에는 그 영향이 여전히 남아 있었다. 현재 Omega와 Lambda는 이미 TEX Live에 포함되지 않게 되었고 Aleph만이 유지되고 있다.

2020년 10월의 LaTeX 개변 이후, ε-TEX의 확장을 Aleph 엔진이 수용할 수 없어서, 현재 Lamed 포맷도 사라진 상태이다. 그러나 Omega의 이상과 그 개발과정에서 남긴 몇 가지 기여는 훗날 유니코드 TEX 시스템의 발전에 영향을 끼쳤다.

테스트용 가상 기계는 Omega와 Aleph가 모두 있다. 포함된 두 예제는 lambda나 lamed를 이용하여 컴파일 가능하다.

5.2.1 HLaTeX의 Omega 지원, HLambda

완성형 제한으로 괴로워하던 HLaTeX은 그 활로를 Omega에서 찾고 있다는 암시를 주었다. 1.0.1 버전은 lamdba에서 돌아가도록 HLaTeX 전체를 보수하였고 실제로 lambda와 UTF-8 한글로써 완성형 밖의 한글과 옛한글을 처리할 수 있음을 입증한 바가 있다.

지금 생각하면 이것은 잘못된 방향이었다. 유니코드 한글은 XeTeX과 LuaTeX의 발전을 기다려서 자연스럽게 해결되었기 때문이다. 그러나, 이 문제로 고군분투하던 당시 개발자들의 고민을 어느 정도 이해해주기를 바란다.

테스트 파일은 `21-HLaTeX-Lambda.tex`이다.

HLaTeX-Lambda HLATEX의 Lambda 지원은, 동작하는 엔진이 Lambda이면 알아서 이루어지는 방식이었다. 그래서 preamble에

```
\usepackage{hangul}
```

만 있어도 충분하였다. 기본 폰트는 (당연하지만) uhc type1이었다.

다음 순서로 테스트해보라.

```
lambda 21-HLaTeX-Lambda.tex
dvipdfm 21-HLaTeX-Lambda.dvi
```

5.2.2 DHHangul

이와는 별도로 Lambda 엔진을 이용하는 거의 새로운 한글 패키지를 김도현이 제작하였다. 이름은 DHHangul. 이것은 앞서 소개한 트루타입 기술 등을 더 진취적으로 활용하여 고품위의 PDF를 제작하는 것을 염두에 두고 만들어진 것이었다. 이 체험판에 포함된 DHHangul은 당시 새로이 조성된 함초롬 LVT 글꼴 트루타입을 디폴트로 하도록 설정하고 있다.

실제로 DHHangul로 제작된 문서 가운데, 〈옛말의 문법〉(2004)이 있었는데 마침내 옛 한글을 완전하게 표현해내게 된 기념비적인 문서였다고 자평한다. (왜냐하면 이거 만든 사람이 필자이다.) 한글LATEX이 2004년 당시 이를 수 있었던 최고 단계를 보여주고 있다. 이 문서에 사용한 폰트는 새바탕 트루타입이었고 한양PUA 방식으로 15세기 한글을 식자하였다.

DHHangul의 테스트용 파일은 `22-DHHangul.tex`인데 단순히 이제까지 예제로 사용해 온 바 yahTEX에서 작성한 문서 그대로이다. 한글 폰트가 어떻게 나타나는가만 확인하자.

```
lamed 22-DHHangul.tex
dvipdfmx 22-DHHangul.dvi
```

이 샘플 문서는 lambda로는 잘 컴파일되지 않으며 lamed로 해야 한다. 그리고 dvipdfm으로는 문제를 만나게 될 것이다. dvipdfmx로 변환하라. 가히 당대 가장 핫한 기술의 총화라고 할 수 있겠다.

5.3 그 이후를 위한 간략한 스케치

이 모든 문제를 거의 대부분 해결한 패키지가 바로 ko.TEX이었다. 그리고 우리는 이것을 더 이상 보충할 것이 별로 없다고 판단하였을 때, 유니코드 지원 부분만을 떼어내어서 CTAN에 업로드하였고, 거의 즉시 한글LATEX의 표준이 되었다.

Omega로 넘어갔던 관심은 새로운 엔진 XETEX에 대응하는 xetexko, LuaTEX에 대응하는 luatexko로 충족되었다. 그리고 오늘날에 이른다. 2007년에서 2013년 사이에 정비된 이 한글LATEX 체계는 부분적인 수정과 발전을 거치면서도 본질적으로 큰 줄기는 변함없이 이어지고 있다. 이 시기의 새로운 발전과 변모 과정은 별도의 글이 필요할 것이다.

6

결론

이 글이 다루는 시기는 대략 1991~2004년이다. 약 13,4년에 이르는 이 시기 동안 한글LaTeX의 변화는 극적이라고 할 만하다.

나는 한글LaTeX의 눈부신 발전이 이루어지던 이 시기의 목격자로서 이 글을 썼다. 제공된 체험판을 운영해보면서 "어떻게 이렇게 불편한 환경에서 살았나"를 궁금해하든가, "뭐 하나 제대로 되는 게 없네"라는 소감을 품든가, 각자의 몫이겠지만 당시 LaTeX 사용자·개발자들이 어떤 요구에 대응하려 하였고 어떤 문제를 해결하려 하였는지 짐작이라도 하게 된다면 좋겠다.

부록 A Sample Document 문서 정보

SampleDocument.zip 파일에 포함되어 있는 출력물(PDF)에 대한 정보. 모든 파일은 yahTeX '마누엘' 문서의 1장을 충실하게 재현한 기본 소스를 본문으로 하고 preamble만을 수정하여 각 환경에서 컴파일되도록 한 것이다. 소스 파일도 포함되어 있다.

eucKR 폴더

- 00-yahtex.pdf yahtex이 생성한 DVI 파일에 대하여 TeX Live 2022의 dvips로 PS, PDF 변환. 폰트는 yahtex 자체가 제공하는 PK bitmat.
- 01-winlatex.pdf winlatex의 DVIWIN에서 가상 Windows의 프린터로 '인쇄'한 결과인 prn 파일을 TeX Live 2022의 dvips로 PS, PDF 변환. 폰트는 winlatex이 300dpi 프린터로 보낸 비트맵.
- 02-HTEX.pdf HTEX에서 가상 windows 98 기계의 프린터로 '인쇄'한 결과인 prn 파일을 TeX Live 2022의 dvips로 PS, PDF 변환. 폰트는 HTEX이 300dpi 가상 포스트스크립트 프린터로 보낸 비트맵. 이 비트맵은 HTEX에 의하여 윈도우즈 바탕, 돋움, 굴림으로부터 생성되었음.
- 03-hltxp_OZ.pdf OzTEX의 DVIPS 기능으로 변환한 PS를 매킨토시 가상 기계 내의 Ghostscript로 PDF 변환. 폰트는 hLaTeXp의 xhan 비트맵 300dpi.
- 10-hltxp.pdf Xubuntu 가상 리눅스에서 hLaTeXp 매크로가 적용된 파일을 dvipdfm 변환. 폰트는 hLaTeXp의 xhan 비트맵 300dpi.
- 11-HLaTeX_dvipdfmx.pdf HLaTeX 예제를 dvipdfmx로 변환. 폰트는 uhc type1.
- 11-HLaTeX_dvips.pdf HLaTeX 예제를 dvips 변환. 폰트는 uhc type1.
- 11-HLaTeX_pdftex.pdf HLaTeX 예제를 pdflatex으로 컴파일.
- 12-hangulk-unttf.pdf dvipdfmx로 변환. 폰트는 은 폰트 트루타입. 이 파일은 텍스트의 검색과 추출이 된다.

UTF8 폴더

- 20-dhucs.pdf dhucs 문서를 dvipdfmx로 컴파일. 폰트는 은 글꼴 트루타입.
- 21-HLaTeX-Lambda.pdf HLaTeX 문서를 lambda로 컴파일, dvipdfm 변환.
- 22-DHHangul.pdf Lamed로 컴파일, dvipdfmx로 변환. HCR LVT 트루타입.
- 30-modernoblivoir.pdf 동일 소스를 oblivoir 클래스로 하고 XeLaTeX으로 컴파일.

부록 B 체험을 위한 소프트웨어

- DOSBOX: version 0.74-3. https://www.dosbox.com/.
- VirtualBox: version 6.1. https://www.virtualbox.org/.
- SheepShaver (압축파일에 내장): https://sheepshaver.cebix.net/.

1. DOSBOX는 Windows, Linux, MacOS용 실행 파일이 있다.

2. VirtualBox는 Windows, Linux에서 설치 가능하나 Mac의 경우는 IntelMac의 경우 문제없이 운용할 수 있다. Apple Silicon Mac은 VirtualBox를 활용할 수 없을 수 있다. VMware Fusion이나 UTM과 같은 다른 대안을 찾아보아야 할지 모른다. 그리고 두 개의 ova가 모두 i386을 기준으로 만들어졌으므로 성공적인 운영을 장담하지 못한다.
3. SheepShaver의 Windows용 바이너리는 압축 파일에 들어 있다. MacOS에서는 E-Maculation 제공 바이너리로 실행가능하다. Linux는 직접 빌드하는 것이 좋다.

부록 C 체험을 위한 파일 목록

- `1992-yahtex-emtex_dosbox.zip` (15.4MB)
- `1994-winlatex-win31_dosbox.zip` (19.4MB)
- `1995-HTEX-win98_vbox.ova` (130MB)
- `1996-OzTeX-Mac755_SheepShaver.exe.zip` (384MB)
- `1998-prekotex-xubuntu16.04_vbox.ova` (4.46GB)
- `SampleDocument.zip` (예제문서)
- `Manuals.zip` yahTeX, WinLaTeX, hLaTeXp, HLaTeX의 안내문서를 pdf 형식으로 모아두었다.

부록 D KTUG 게시판의 [유물] 관련글

- 〈[취미] DOSBOX에서 WinLaTeX〉 (noname, 2021/07/02)
- 〈[유물] 1995년의 TeX: HTEX〉 (noname, 2022/04/17)
- 〈[유물] 1996년의 TeX: 매킨토시 OzTeX+hLaTeXp〉 (noname, 2022/04/20)