TL/1 언어의 활용

애플의 언어

대부분의 애플 사용자들은 베이직 언어로 프로그램을 작성하고 있으나 베이직 고유의 여러가지 단점때문에 많은 고충을 겪고 있다. 특히 견디기 어려운 것은 엄청나게 느린 속도이다. 이 느린 속도를 극복하기 위해서 베이직 컴파일러를 사용한다고 해도 느린 속도가 조금 극복되었을 뿐 또 다른 애로점이 기다리고 있다. 더구나 컴파일시킨 프로그램은 디스켓의 많은 메모리를 차지하고 있어서 괜시리 디스켓이 너무 낭비되지 않나 하는 걱정이 앞선다. 필자는 이러한 고충을 안고 베이직을 사용하던 중 TL/1이란 언어를 알게 되었고, 차츰 그 속도와 프로그램의 작성효율에 매료되기 시작하였다.

일단 애플소프트 베이직, 컴파일시킨 베이직 프로그램, 기계어(6502), 터보 파스칼, TL/1 언어를 수행속도와 프로그램의 효율성에 맞춰 비교해보자. 표 1은 이 언어들의 비교표를 만든 것이다.

표 1을 보면 알 수 있듯이 베이직 언어는 배우기가 쉽고 여러 사용자들에 의해서 널리 쓰인다는 장점이 있지만, 속도가 너무 느리고 프로그램을 도와주는 환경이 미약할 뿐만 아니라 구조적인 프로그램을 작성하기가 힘들다는 단점이 있다. 이 베이직 프로그램을 컴파일해서 사용할 경우에는 3~5배 정도로 속도를 향상시킬 수 있으나 런타임 라이브러리가 첨가되어 화일의 크기가 커지는 단점이 있다. 이에 반해 기계어는 하드웨어의 성능을 최대로 이용할 수 있고 속도가 엄청나게 빠르다는 장점이 있으나 웬만한 실력으로는 프로그램을 제대로 작성하기 힘들다는 단점이 있다.

꼬마 컴파일러 언어인 TL/1. 작은 몸집을 가지고 큰 힘을 발휘해서 우리를 신선한 충격속으로 몰아 넣었던 이 언어를 마음껏 사용하고 싶지 않습니까? 여기 소개하는 TL/1으로 작성한 여러개의 예제 프로그램과 다른 언어와의 수행속도, 작업환경, 구조화 프로그래밍 작성 효율의 비교를 통해 7K의 꼬마를 마음껏 조정해 보기 바랍니다.

글·이 찬 진

터보 파스칼은 빠른 속도, 좋은 프로그램 환경, 구조적 프로그램 등의 장점을 골고루 갖추고 있으나 아쉽게도 CP/M상에서 운용되는 관계로 웬지 낯선 느낌이 들고, 애플이 가지는 하드웨어적인 특성들을 제대로 활용하기가 쉽지 않다. 예를 들면 애플 베이직에서는 당연히 돌아가는 고해상도 그래픽을 터보 파스칼에서 쓰기 위해서는 TURBO HGR과 같은 별도의 프로시듀어를 사용해야 하는 불편이 따른다.

이렇게 애플에서 돌아가는 언어들의 특성상 주 프로그램은 베이직으로 작성하고, 시간을 많이 필요로 하는 부분은 기계어 서브루틴으로 작성하는 방법이 바람직하다는 결론을 얻었다. 필자는 이를 좀더 발전시켜 기계어 서브루틴을 TL/1으로 대치하는 것이 좋다는 생각에 이르렀다. 물론 속도는 기계어보다 3~5배정도 느리지만 베이직보다는 20~50배 정도가 빠르므로 필요한 속도는 충분히 낼 수 있고 무엇보다도 프로그램을 작성하기 쉽다는 장점이 있으니까.

TL/1 과 파스칼

파스칼을 흉내내서 TL/1이란 언어를 만들었는지는 자세히 모르겠지만 이 2가지 언어는 흡사한 형태를 가지고 있다. 일단 파스칼과 TL/1의 차이점을 알아보면 다음과 같다.

① 파스칼의 begin~end 문은 TL/1에서 [ ], { }, ( ) 등으로 대치될 수 있다. 물론 begin~end 문을 그대로 사용해도 된다. 여기서 주의할 점은 TL/1의 주 프로그램과 프로시듀어의 시작과 끝에 있는 한쌍의 begin~end는 [ ], { }, ( ) 등으로 대치될 수 없다는 것이다.

② 파스칼에서 문장을 구별하는 ';'은 TL/1에서 사용하지 않아도 된다. 물론 사용한다고 해도 지장은 없지만 TL/1에서의 ;는 아무런 의미를 갖지 못한다. 이에 대해서는 예제 1을 참고하기 바란다.

③ TL/1은 프로시듀어의 인수전달에 Call by Value만 허용하고 Call by Reference는 허용되지 않는다. 따라서 Call by Reference가 필요할 때는 전역 변수(Global Variable)를 사용해야 한다.

표 1 : 수행속도와 작업효율에 따른 비교

	1	×	×	메인 프로그램	너무 느리다.
	3~5배	×	×	베이직 프로그램의 속도를 개선할 때 사용한다.	크기가 커진다.
	100~150배	×	△	속도가 요구되는 일을 맡는 서브루틴	프로그램하기가 어렵다.
	20~50배	○	○	CP/M 환경하의 모든 일	CP/M에서 운영되므로 애플 도스 사용자에게는 불편함을 준다.
	30~60배	○	×	기계어와 같은 용도로서 기계어에 익숙하지 않은 프로그래머가 사용하면 좋다.	작업 환경이 미약하다.

예제 1 : TL/1과 파스칼의 문장구조 비교

```
VAR   I, J                          VAR I, J : INTEGER;
BEGIN                               BEGIN
  FOR I : =1 TO 10 DO {               FOR I : =1 TO 10 DO
      FOR J : =1 TO 10 DO {             BEGIN
          WRITE (0 : I, CRLF)}            FOR J : =1 TO 10 DO
          WRITE (0 : CRLF)}                 WRITELN
END                                       END
                                    END.
FOR I : =1 TO 10 DO { }             FOR I : =1 TO 10 DO;
```

예제 2 : TL/1과 파스칼의 프로시듀어 정의방법 비교

```
PROC A                              Procedure A(I, J) ;
FUNC B                              begin
VAR   I, J, K                         write (I, J)
BEGIN                               end;
  I : =1                            Function   B(I, J) ;
  J : =2                            begin
  A(I, J)                             B : =I+J
  K : =B(I, J)                      end;
END
                                    begin
  A(I, J)                             I : =1;
  BEGIN                               J : =2;
    WRITE (0 : I, J )                 A(I, J) ;
  END                                 K : =B(I, J)
                                    end.
  B(I, J)
  BEGIN
    RETURN I+J
END
```

④ 펑션이나 프로시듀어를 정의하는 방법에는 차이가 있다. (예제 2 참고)

⑤ TL/1의 case 문은 문법이 다르다. 예제 3은 이를 비교해 놓은 것이다.

⑥ TL/1에는 바이트 변수만 있으므로 변수의 선언도 그 형태가 다르다.

리스트 1의 프로그램은 TL/1을 최대한 파스칼과 닮게 작성한 것이므로 파스칼과 TL/1을 비교하는데 참고하기 바란다.

TL/1과 기계어

TL/1의 특징을 요약해보면, 파스칼의 문법구조를 가지고 있고 실제로 수행되는 수행문들은 기계어적인 특성을 가진다고 말할 수 있다. 파스칼적인 문법구조는 앞에서 이미 설명했고, 기계어적인 특성은 다음과 같다.

① TL/1의 데이터들은 모두 1바이트로 구성되어 있다. 이 1바이트는 10진수, 16진수, 문자, 논리값의 형태로 쓰이는데 다음의 문장은 모두 A에 65라는 값이 대응된다.

```
A : =65 (10진수)
A : =$41 (16진수)
A : ='A' (문자 A의 아스키 값은 65임)
```

그리고 TRUE는 255($FF), FALSE는 0의 값을 가진다.

예제 3 : TL/1과 파스칼의 case문 비교

CASE A OF 1 B : =A 2 {B : =A+1 WRITE (0 : CRLF)} 3 B : =A+2 ELSE B : =A+3	case A of 1 : B : =A 2 : begin B : =A+1; writeln end 3 : B : =A+2 4..9 : B : =A+3

②1바이트 연산이나 처리명령중에는 기계어와 똑같은 명령이 많다. 예를 들면 ADC, SBC, ASL, ASR, LSR, ROL, ROR 등이다.

③1바이트 변수인 관계로 덧셈이나 곱셈의 결과가 1바이트를 넘는 경우가 생기므로 주의해야 한다. 필자의 경험으로는 TL/1으로 프로그램을 작성할 때 버그의 80% 이상이 이 부분에서 발생했다.

계산결과가 255를 넘을 경우에는 변수를 상위 바이트와 하위 바이트로 나누어서 작성해야 하는데, 다음은 틀린 예와 맞는 예를 보인 것이다.

A : =100 B : =200 C : =A+B	A : =100 B : =200 CL : =A+B CH : = (0, ADC, 0)
D : =A*B	DL : =A*B DH : =MHIGH

이 예에서 MHIGH는 곱셈결과의 상위 바이트를 저장하고 있는 시스템 변수이다.

④나머지를 구하는 연산은 다음과 같다. (C는 몫이고 D는 나머지이다. 여기서 MOD는 나머지값을 기억하는 시스템 변수이다.)

```
A : =100
B : =3
C : =A/B
D : =MOD
```

⑤TL/1에서는 기계어 루틴을 사용하기가 쉽다. 사용 형식은 다음과 같다.

```
CALL (AH, AL, A, X, Y)
USR (AH, AL, A, X, Y)
```

여기서 AH와 AL은 호출하기를 원하는 서브루틴의 상위와 하위 바이트이고, A, X, Y는 어큐뮬레이터, X레지스터, Y레지스터의 값을 설정할 때 사용된다. 그리고 CALL은 프로시듀어이고, USR은 평션으로 돌아올 때 어큐뮬레이터의 값을 가지고 돌아온다. 리스트 2는 CALL 문을 사용한 예로서 고해상도 그래픽 명령을 TL/1에서 사용할 수 있게 작성한 것이다.

베이직과의 비교

이제부터는 몇개의 예제 프로그램을 통해서 TL/1과 베이직을 비교해보자. 이제부터 이 2가지 언어로 작성된 프로그램을 통해서 TL/1을 익히고 속도를 비교해 보겠다.

(1) DUMP

리스트 3과 4는 메모리의 내용을 16진수와 아스키 코드로 덤프하는 프로그램이다. 이 2개의 프로그램을 비교해 보면 알겠지만 베이직은 16진수로 진법변환 루틴을 추가해야 하지만 TL/1에서는 HEX( )라는 출력함수가 마련되어 있다. 이 프로그램은 화면출력 과정이 삽입되어 속도차이는 많이 나지 않았다.

(2) QUIZ

리스트 5와 6은 컴퓨터로 풀어보는 퀴즈다. 내용은 수탉이 50원, 암탉이 30원, 병아리가 3마리에 10원씩할 때 이 세가지를 섞어서 100마리에 1000원이 되는 경우를 찾는 프로그램이다. 리스트 5의 TL/1 프로그램에서는 ADC를 이용해서 2바이트 덧셈을 하고 있으므로 눈여겨 보기 바란다.

(3) LOOP

리스트 7과 8은 빈 3중 루프를 $100^3 - 1,000,000$ (백만번) 도는 프로그램이다. 속도를 비교해 보기 바란다.

(4) INVERT

리스트 9와 10은 고해상도 1페이지의 그림을 역상으로 만들어 주는 프로그램이다. TL/1의 함수인 COM을 이용하고 있으니 주의깊게 살펴보기 바란다.

(5) ROM-CONV

리스트 11은 지난 4월호에 게재된 '완벽한 애플 키보드를 향하여' 기사중 '데이터 변환용 프로그램'을 TL/1으로 변환시킨 것이다. 이 프로그램은 $3F00 번지의 값에 따라 2가지 변환중에 하나를 수행한다. 이 프로그램내의 10진수와 2진수의 진법변환 프로시듀어는 도움이 많이 되니 참고하기 바란다.

(6) HGR-CONV

리스트 12는 지난 8월호와 9월호에 소개되었던 '그래픽 컨버터'를 TL/1으로 다시 작성한 것이다. 8월호에 실린 프로그램이 30분, 9월호에 실린 프로그램이 10분 정도의 수행시간을 가졌던 것에 비하면 불과 65초 정도에 변환과 출력을 완료시켜 준다.

리스트 12는 변환기능을 가진 TL/1 프로그램이고, 리스트 13은 속도 비교를 위해 같은 알고리즘으로 작성한 베이직 프로그램이다.

(7) SCROLL

리스트 14와 15는 텍스트 화면과 HGR화면을 4방향으로 스크롤시키는 프로그램이다.

(8) 속도 비교결과

표 2는 앞에서 설명한 6개의 프로그램의 속도를 비교해 놓은 것이다.

표 2 : 예제 프로그램의 속도 비교

DUMP	197	21		9.4
QUIZ	705	22	35	32.0
LOOP	1,395	21	46	66.4
INVERT	60	0.7	0.9	85.7
HGR-CONV	4,100	65		63.1
ROM-CONV	3,000	36		83.3

TL/1 사용상의 도움말

①TL/1도 베이직과 마찬가지로 도스명령을 프로그램내에서 사용할 수 있다.

그 방법은 다음과 같다.

```
WRITE (0 : "CATALOG")
        ⊂(Ctrl D)를 누른다.
WRITE (0 : ASCII (13), "CATALOG")
```

②리스트 2의 HGR 프로시듀어처럼 인터프리터나 모니터 루틴을 CALL이나 USR 문을 사용하여 라이브러리로 만들어 놓고 다른 프로그램과 합해서 사용하면 유용할 것이다.

③런타임 라이브러리는 1페이지 단위의 번지(예를 들면 $8800이나 $9000)에 BLOAD하고, 그 시작번지를 CALL하면 재배치할 수 있다. 이때 TL/1이라는 화일이 메모리내에 있어야 한다.

리스트 16의 베이직 프로그램은 라이브러리를 $9000으로 재배치하는 HELLO 프로그램이다.

④변수영역은 라이브러리의 바로 뒤에 위치하지만 필요에 따라서는 $980에 하위번지를, $982에 상위번지를 입력함으로써 변수영역을 변경시킬 수 있다. 이렇게 변수영역을 변경시키는 작업도 TL/1이 실행된 후에 해야 한다.

⑤TL/1 프로그램을 베이직 프로그램과 연결시키기 위해서는 컴파일한 오브젝트 화일을 따로 BSAVE시켜야 하는데 자세한 사항은 '베이직과 TL/1의 연결'을 참조하기 바란다.

베이직과 TL/1의 연결

TL/1으로 작성한 프로그램을 베이직으로 작성된 주프로그램과 연결시키기 위해서는 약간의 절차가 필요하다. 설명을 쉽게하기 위해 '그래픽 컨버터'를 예로 들겠다.

리스트 17은 9월호에 소개한 알고리즘을 이용하여 리스트 12의 프로그램을 개선한 것으로서 변환시간이 26초로 단축되었다.

우선 TL/1을 시동해서 리스트 17을 컴파일하면,

```
COMPILATION COMPLETE.
OBJECT    RUNS    AT   $9200
OBJECT    BEGINS  AT   $2697
OBJECT    ENDS    AT   $298F
VARIABLE  BASE    AT   $9500
```

이라는 메시지가 출력될 것이다. 이것은 리스트 16의 HELLO 프로그램으로 라이브러리(TL2)를 $9000 번지에 재배치한 경우이다. 이때 메모리 맵은 그림 1과 같다. 여기서 필요한 부분은 컴파일된 오브젝트와 라이브러리이다. 먼저 다음의 명령으로 실제 라이브러리 부분을 디스켓에 저장한다.

```
BSAVE LIBRARY, A$9200, L$300
```

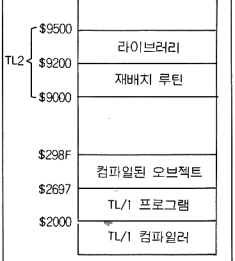
또한 오브젝트는 PROGRAM이라는 이름으로 저장해야 하는데, 그 전에 약간의 수정이 필요하다. 즉, TL/1 프로그램이 컴파일러와 함께 있으면서 실행될 때는 프로그램이 끝나면 TL/1에디터의 웜 스타트 번지($1C03)로 점프하는데, 베이직과 함께 쓸 경우에는 이 값을 $D823으로 바꿔줘야 한다. 'JMP $1C03' 명령의 위치는 TL/1 주프로그램의 끝에 위치하거나 프로시듀어의 정의 직전인데, 이 경우에는 $28CE번지에 있다. 이제 $28CE번지에 있는 'JMP $1C03'을 'JMP $D823'으로 바꾼 후에 다음과 같은 명령을 내리면 모든 준비는 끝난 것이다.

```
BSAVE PROGRAM A$2697, L$2F8
```

이제 리스트 18의 베이직 프로그램을 입력해서(TL/1을 사용중이었다면 부팅시킨 후에) 다음과 같이 디스켓에 저장한 후에 실행시키면 베이직과 TL/1은 연결될 것이다.

```
SAVE HGR CONVERTER
```

그림 1 : 라이브러리가 재배치된후의 메모리 맵



한편 이 프로그램은 엡슨계열의 LX-80이나 LX-86 프린터를 기준으로 해서 제어코드를 사용했다. 특히 60~70행에서는 엘리트-축소모드를 사용해서 160자까지 찍을 수 있도록 만들었고, 80행에서는 라인 스페이싱을 조절하였다. 그리고 HGR화면중 일부만 찍을 수 있도록 90행에서 프린트되는 범위를 지정했는데 현재는 0~139 컬럼(140자), 0~63라인을 프린트한다. 엡슨 프린터의 다른 모드나 다른 프린터를 사용하는 독자들은 이 값을 적절히 조정하기 바란다.

TL/1이 가지는 문제점과 개선점

(1) 에디터

현재 TL/1이 갖고 있는 가장 큰 문제점은 에디터이다. TL/1을 사용해본 독자들은 알겠지만 리스트 명령의 범위를 지정할 수 없기 때문에 제일 끝행을 보려면 처음부터 끝까지 모두 리스트해야 한다. 따라서 프로그램이 조금만 길어지면 그 불편함은 더욱 커지게 된다. 또한 여러 행을 지울 때에는 삭제하려는 모든 행번호를 일일이 입력하여 한 행씩 지워야 하는 불편이 따른다.

(2) 카탈로그 출력정지와 도스명령 사용

TL/1은 프로그램의 로드나 세이브시에 화일의 카탈로그가 표시되어 불편한데, 다음과 같이 고치면 이 문제를 쉽게 해결할 수 있다.

*IBEA : EA EA EA〈RETURN〉

그리고 M명령으로 모니터 모드를 빠져 나와서 베이직 상태로 돌아온 후에 도스명령(CATALOG)을 한번 수행하고 'CALL -151〈RETURN〉' 명령으로 모니터 모드로 간다. 그런후에 〈Ctrl Y〉 명령으로 TL/1의 에디터 모드로 들어간다. 그 이후로는 에디터 모드에서 도스명령을 사용할 수 있다.

(3) 리스트 출력의 개선

TL/1은 컴파일시에 리스트가 출력되어 속도가 늦어진다. 이때는 리스트를 출력하지 않고 에러가 발생했을 경우에만 에러 메시지와 행번호만을 출력하도록 TL/1을 개선해야 한다. 또한 TL/1은 프로그램이 실행될 때 변수를 초기화하지 않아서 혼란의 소지가 있다.

(4) 'OUT OF BUFFER' 에러 해결방법

가끔씩 'OUT OF BUFFER' 라는 에러가 발생하는데, 이 경우에는 다음과 같이 해결한다.

① ':M〈RETURN〉' 명령으로 모니터 모드로 들어간다.

② 〈Ctrl C〉 명령으로 베이직 모드로 나온다.

③ 'FP〈RETURN〉' 명령을 내린다.

④ 'CALL-151〈RETURN〉' 명령으로 모니터 모드로 간다.

⑤ 〈Ctrl Y〉 명령으로 TL/1의 에디터 모드로 간다.

이와 같이 하면 'OUT OF BUFFER' 에러가 해결될 것이다.

마무리 짓는 말

TL/1을 혼자만 꼭꼭 숨어서 사용하기가 아깝다는 생각에 이것 저것 순서없이 늘어 놓았는데, 독자 여러분에게 혼란을 주지 않았는지 걱정이 앞선다. 하지만 TL/1에 대해 조금이라도 관심이 모아졌다면 글을 쓴 보람이 있었다고 자위하며 이 글을 마치고자 한다. 더불어 TL/1의 버그를 잡고 쓸만한 에디터를 달아주실 분이 계셨으면 하는 소망과 시간이 나면 TL/1으로 중앙한 글의 에디터를 만들어 발표하겠다는 희미한 약속을 드린다.

리스트 1 : TL/1을 파스칼 형식으로 작성한 예

```
1000
1010 FUNC
1020   HSCRN;
1030 VAR
1040    I,J,X1,Y1,X2,Y2,XH,XL,Y,SUM,CHAR;
1050 BEGIN
1060    MEM($00,$E6):=$40;
1070    X1:=0;
1080    Y1:=0;
1090    X2:=40;
1100    Y2:=63;
1110    FOR I:=Y1 TO Y2 DO
1120      BEGIN
1130        Y:=I*3;
1140        FOR J:=X1 TO X2 DO
1150          BEGIN
1160            XL:=J*2;
1170            XH:=MHIGH;
```

1

```
1180            SUM:= HSCRN(XH,XL  ,Y  )
1190                 +HSCRN(XH,XL+1,Y  )
1200                 +HSCRN(XH,XL  ,Y+1)
1210                 +HSCRN(XH,XL+1,Y+1)
1220                 +HSCRN(XH,XL  ,Y+2)
1230                 +HSCRN(XH,XL+1,Y+2);
1240            CASE SUM OF
1250               0 CHAR:=' '
1260               5 CHAR:='.'
1270               4 CHAR:='-'
1280               3 CHAR:='+'
1290               2 CHAR:='9'
1300               1 CHAR:='*'
1310            ELSE CHAR:='B';
1320            WRITE(0:ASCII(CHAR))
1330          END;
1340        WRITE(0:CRLF)
1350    END
```

2

```
1360 END.
1370 %
1380 % HGR-SCRN ROUTINE
1390 %
1400 HSCRN(XH,XL,Y);
1430 BEGIN
1570   RETURN Y
1580 END
```

리스트 2 : HGR.TL

```
1000 PROC
1010   HGR,HCOLOR,HPLOT,HPLOTTO,HLINE
1020 BEGIN
1030   HGR
1040   HCOLOR(3)
1050   HPLOT(0,0,191)
1060   HPLOTTO(1,23,0)
1070   HLINE(1,23,191,0,0,0)
1080   HPLOTTO(1,23,0)
1090   HPLOTTO(1,23,191)
1100   HPLOTTO(0,0,191)
1110   HPLOTTO(0,0,0)
1120 END
1130 %-------------------------
1140 HGR
1150 BEGIN
1160   CALL($F3,$D8)
1170 END
1180 %-------------------------
1190 HCOLOR(COLOR)
1200 BEGIN
1210   CALL($F6,$F0,0,COLOR)
1220 END
1230 %-------------------------
1240 HPLOT(XH,XL,Y)
1250 BEGIN
1260   CALL($F4,$57,Y,XL,XH)
1270 END
1280 %-------------------------
1290 HPLOTTO(XH,XL,Y)
1300 BEGIN
1310   CALL($F5,$3A,XL,XH,Y)
1320 END
1330 %-------------------------
1340 HLINE(XH1,XL1,Y1,XH2,XL2,Y2)
1350 BEGIN
1360   HPLOT(XH1,XL1,Y1)
1370   HPLOTTO(XH2,XL2,Y2)
1380 END
```

리스트 3 : TL/1으로 작성한 덤프 프로그램

```
1000 VAR
1010   I,J,K,START,STOP,DATA
1020 BEGIN
1030   START:=$40
1040   STOP:=$4F
1050   FOR I:=START TO STOP DO
1060     FOR J:=$0 TO $F DO [
1070       WRITE(0:HEX(I),HEX($10*J)," : ")
1080       FOR K:=0 TO $F DO
1090         WRITE(0:HEX(MEM(I,J*$10+K))," ")
1100       WRITE(0:": ")
1110       FOR K:=0 TO $F DO [
1120         DATA:=MEM(I,J*$10+K)AND$7F
1130         IF DATA<$20 THEN DATA:=$20
1140         WRITE(0:ASCII(DATA)) ]
1150       WRITE(0:CRLF)
1160       SENSE ]
1170 END
```

리스트 4 : 베이직으로 작성한 덤프 프로그램

```
100 ST = 4 * 4096
110 EN = 5 * 4096 - 1
115 :
120   DIM H$(15)
130   FOR I = 0 TO 15
140   READ A$
150 H$(I) = A$
160   NEXT I
170   DATA 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
175 :
180   FOR I = ST TO EN STEP 16
190 D = I: GOSUB 400
200   FOR J = 0 TO 15
210 D =  PEEK (I + J): GOSUB 500
220   NEXT J
230   PRINT ": ";
240   FOR J = 0 TO 15
250 D =  PEEK (I + J)
260   IF D < 160 THEN  PRINT " ";: GOTO 280
270   PRINT  CHR$ (D);
280   NEXT J
290   PRINT
300   NEXT I
310   END
320 :
400 A% = D / 4096:D = D - (4096 * A%)
410 B% = D / 256:D = D - (256 * B%)
420 C% = D / 16:D = D - (16 * C%)
430 D% = D
440   PRINT H$(A%);H$(B%);H$(C%);H$(D%);" : ";
450   RETURN
460 :
500 A% = D / 16:D = D - (16 * A%)
510 B% = D
520   PRINT H$(A%);H$(B%);" ";
530   RETURN
```

리스트 5 : TL/1으로 작성한 퀴즈 프로그램

```
1000 VAR I,J,K,SH,SL,M
1010 BEGIN
1020   FOR I:=0 TO 20 DO [
1030     FOR J:=0 TO 33 DO
1040       FOR K:=0 TO 33 DO [
1050         SL:=0          SH:=0
1060         SL:=SL+I*50 SH:=SH.ADC.MHIGH
1070         SL:=SL+J*30 SH:=SH.ADC.MHIGH
1080         SL:=SL+K*10 SH:=SH.ADC.MHIGH
1090         M:=I+J+K*3
1100         IF SL=$E8 AND SH=3 AND M=100
1110           THEN WRITE(0:I," ",J," ",K*3) ]
1120     WRITE(0:".",CRLF) ]
1130 END
```

리스트 6 : 베이직으로 작성한 퀴즈 프로그램

```
1  FOR I = 0 TO 20
2  FOR J = 0 TO 33
3  FOR K = 0 TO 33
4  SUM = I * 50 + J * 30 + K * 10
5  M = I + J + K * 3
6  IF SUM = 1000 AND M = 100 THEN  PRINT I" "J"
     "K * 3
7  NEXT K,J: PRINT ".": NEXT I
```

리스트 7 : TL/1으로 작성한 루프 프로그램

```
1000 VAR I,J,K
1010 BEGIN
1020   FOR I:=1 TO 100 DO {
1030     FOR J:=1 TO 100 DO {
1040       FOR K:=1 TO 100 DO { }}
1050     WRITE(0:".") }
1060 END
```

1

2.

3.

리스트 8 : 베이직으로 작성한 루프 프로그램

```
1  FOR I = 1 TO 100
2  FOR J = 1 TO 100
3  FOR K = 1 TO 100
4  NEXT K,J: PRINT ".";: NEXT I
```

리스트 9 : TL/1으로 작성한 역상 프로그램

```
1000 VAR I,J
1010 BEGIN
1020   MEM($C0,$50):=0
1030   MEM($C0,$57):=0
1040   MEM($C0,$55):=0
1050   MEM($C0,$52):=0
1060   FOR I:=$40 TO $5F DO
1070     FOR J:=0 TO $FF DO
1080       MEM(I,J):=COM(MEM(I,J))
1090 END
```

리스트 10 : 베이직으로 작성한 역상 프로그램

```
1  HGR2
2  FOR I = 16384 TO 24575
3  POKE I,255 -  PEEK (I)
4  NEXT I
```

리스트 11 : 롬 데이터 변환용 프로그램

```
1000 %============================
1010 %
1020 % ROM DATA CONVERTOR
1030 %
1040 % 1987.8.21  BY LEE CHAN JIN
1050 %
1060 %============================
1070
1080 PROC
1090   ROMMEM,MEMROM,BIN,DEC,SHOW
1100 VAR
1110   I,J,Q,W,R,T
1120 ARRAY
1130   A[7],B[7]
1140 BEGIN
1150   IF MEM($3F,$00)=0
1160     THEN ROMMEM
1170     ELSE MEMROM
1180 END
1190 %
1200 % ROM -> MEM
1210 %
1220 ROMMEM
1230 BEGIN
1240   FOR I:=0 TO $F DO [
1250     FOR J:=0 TO $FF DO [
1260       BIN(MEM($40+I,J))
1270       B[0]:=A[4] B[1]:=A[3]
1280       B[2]:=A[6] B[3]:=A[1]
1290       B[4]:=A[2] B[5]:=A[0]
1300       B[6]:=A[7] B[7]:=A[5]
1310       DEC
1320       MEM($40+I,J):=W ]
1330     SHOW ]
1340   WRITE(0:CRLF)
1350   FOR I:=0 TO 7 DO B[I]:=0
1360   FOR I:=0 TO $F DO [
1370     FOR J:=0 TO $FF DO [
1380       BIN(I)
1390       R:=A[0]
1400       B[0]:=A[1] B[1]:=A[3]
1410       B[2]:=A[2]
1420       Q:=J/2      B[3]:=MOD
1430       DEC
1440       T:=LSR(J)+R*128
1450       MEM($50+W,T):=MEM($40+I,J)]
1460     SHOW ]
1470 END
1480 %
1490 % MEM -> ROM
1500 %
1510 MEMROM
1520 BEGIN
1530   FOR I:=0 TO $F DO [
1540     FOR J:=0 TO $FF DO [
1550       BIN(MEM($40+I,J))
1560       B[0]:=A[5] B[1]:=A[3]
1570       B[2]:=A[4] B[3]:=A[1]
1580       B[4]:=A[0] B[5]:=A[7]
1590       B[6]:=A[2] B[7]:=A[6]
1600       DEC
1610       MEM($40+I,J):=W ]
1620     SHOW ]
1630   WRITE(0:CRLF)
1640   FOR I:=0 TO 7 DO B[I]:=0
1650   FOR I:=0 TO $F DO [
1660     FOR J:=0 TO $FF DO [
1670       BIN(I)
1680       R:=A[3]
1690       B[1]:=A[0] B[2]:=A[2]
1700       B[3]:=A[1]
1710       B[0]:=(J>127)/255
1720       DEC
1730       T:=ASL(J)+R
1740       MEM($50+W,T):=MEM($40+I,J)]
1750     SHOW ]
1760 END
1770 %
1780 % DEC -> BIN
1790 %
1800 BIN(IN)
1810 VAR I
1820 BEGIN
1830   FOR I:=7 DOWNTO 0 DO [
1840     A[I]:=IN/128
1850     IN:=ASL(IN) ]
1860 END
1870 %
1880 % BIN -> DEC
1890 %
1900 DEC
1910 VAR I,J,K
1920 BEGIN
1930   W:=0
1940   FOR I:=0 TO 7 DO [
1950     K:=1
1960     IF I#0 THEN
1970       FOR J:=1 TO I DO
1980         K:=ASL(K)
1990     W:=W+B[I]*K ]
2000 END
2010 %
2020 % SHOW PROCESS
2030 %
2040 SHOW
2050 BEGIN
2060   WRITE(0:".")
2070 END
```

리스트 12 : TL/1으로 작성한 그래픽 컨버터

```
1000 %=========================
1010 %
1020 % THIS PROGRAM CONVERTS
1030 % HGR IMAGE TO CHAR PLOT
1040 %
1050 %=========================
1130
1140 FUNC
1150   HSCRN
1160 VAR
1170   I,J,XH,XL,Y,SUM,CHAR
1180 BEGIN
```

```
1190   MEM($Ø0,$E6):=$4Ø
1200   FOR I:=Ø TO 63 DO [
1210     Y:=I*3
1220     FOR J:=Ø TO 139 DO [
1230       XL:=J*2
1240       XH:=MHIGH
1250       SUM:= HSCRN(XH,XL  ,Y  )
1260           +HSCRN(XH,XL+1,Y  )
1270           +HSCRN(XH,XL  ,Y+1)
1280           +HSCRN(XH,XL+1,Y+1)
1290           +HSCRN(XH,XL  ,Y+2)
1300           +HSCRN(XH,XL+1,Y+2)
1310       CASE SUM OF
1320          6 CHAR:=' '
1330          5 CHAR:='.'
1340          4 CHAR:='-'
1350          3 CHAR:='+'
1360          2 CHAR:='?'
1370          1 CHAR:='*'
1380       ELSE CHAR:='B'
1390       WRITE(Ø:ASCII(CHAR)) ]
1400     WRITE(Ø:CRLF) ]
1410 END
1420 %
1430 % HGR-SCRN ROUTINE
1440 %
1450 HSCRN(XH,XL,Y)
1460 VAR
1470   BASEH,BASEL,BYTE,BIT,DATA,I,OUT
1480 BEGIN
1490   CALL($F4,$11,Y)
1500   BASEH:=MEM($Ø0,$27)
1510   BASEL:=MEM($Ø0,$26)
1520   IF XH=Ø
1530     THEN BYTE:=XL/7
1540     ELSE BYTE:=36+(XL+4)/7
1550   BIT:=MOD
1560   DATA:=MEM(BASEH,BASEL+BYTE)
1570   FOR I:=Ø TO BIT DO [
1580     DATA:=LSR(DATA)
1590     OUT:=Ø.ADC.Ø ]
1600   RETURN OUT
1610 END
```

2.

리스트 13 : 베이직으로 작성한 그래픽 컨버터

```
1  DIM B(191)
2  FOR I = Ø TO 6: READ A$(I): NEXT
3  DATA B,*,?,+,-,.," "
4  FOR I = Ø TO 6: READ X: POKE 768 + I,X: NEXT

5  DATA 32,63,255,32,17,244,96
6  FOR I = Ø TO 191: POKE 69,I: CALL 768:B(I) =
     PEEK (38) +  PEEK (39) * 256: NEXT
10  POKE 230,64
11  PRINT "."
20  FOR I = Ø TO 63
40  FOR J = Ø TO 139
41 SUM = Ø
42  FOR K = Ø TO 2
43 Y = I * 3 + K
45  FOR L = Ø TO 1
46 X = J * 2 + L
50 BYTE =  INT (X / 7)
51 BIT = X - BYTE * 7
52 DOT =  PEEK (B(Y) + BYTE)
53 DOT = DOT / 2 ^ (BIT + 1)
54 DOT = DOT -  INT (DOT)
55 Z = (DOT >  = .5)
90 SUM = SUM + Z
100  NEXT L,K
110  PRINT A$(SUM);
160  NEXT J: PRINT : NEXT I
190  END
```

리스트 14 : 텍스트 화면 스크롤 프로그램

```
1000 PROC
1010   UP,DOWN,LEFT,RIGHT
1020 VAR
1030   I,X1,Y1,X2,Y2,TEMP
1040 ARRAY
1050   LINE[39]
1060 BEGIN
1070   FOR I:=Ø TO 191 DO [
1080     CALL($F4,$11,I)
1090     MEM($2E,I):=MEM($Ø0,$27)
1100     MEM($2F,I):=MEM($Ø0,$26) ]
1110   X1:=1Ø
1120   Y1:=6Ø
1130   X2:=3Ø
1140   Y2:=13Ø
1150   MEM($Ø0,$E6):=$4Ø
1160   MEM($CØ,$5Ø):=Ø
1170   MEM($CØ,$57):=Ø
1180   MEM($CØ,$55):=Ø
1190   FOR I:=1 TO X2-X1+1 DO RIGHT
1200   FOR I:=1 TO X2-X1+1 DO LEFT
1210   FOR I:=1 TO Y2-Y1+1 DO UP
1220   FOR I:=1 TO Y2-Y1+1 DO DOWN
1230   MEM($CØ,$51):=Ø
1240   MEM($CØ,$54):=Ø
1250 END
1260 %-----------------------------
1270 UP
1280 VAR I,J,L,M,N,O
1290 BEGIN
1300   FOR I:=X1 TO X2 DO
1310     LINE[I]:=MEM(MEM($2E,Y1),MEM($2F,Y1)+I)
1320   FOR I:=Y1 TO Y2-1 DO [
1330     L:=MEM($2E,I)
1340     M:=MEM($2F,I)
1350     N:=MEM($2E,I+1)
1360     O:=MEM($2F,I+1)
1370     FOR J:=X1 TO X2 DO
1380       MEM(L,M+J):=MEM(N,O+J) ]
1390   FOR I:=X1 TO X2 DO
1400     MEM(MEM($2E,Y2),MEM($2F,Y2)+I):=LINE[I]
1410 END
1420 %-----------------------------
1430 DOWN
1440 VAR I,J,L,M,N,O
1450 BEGIN
1460   FOR I:=X1 TO X2 DO
1470     LINE[I]:=MEM(MEM($2E,Y2),MEM($2F,Y2)+I)
1480   FOR I:=Y2 DOWNTO Y1+1 DO [
1490     L:=MEM($2E,I)
1500     M:=MEM($2F,I)
1510     N:=MEM($2E,I-1)
1520     O:=MEM($2F,I-1)
1530     FOR J:=X1 TO X2 DO
1540       MEM(L,M+J):=MEM(N,O+J) ]
1550   FOR I:=X1 TO X2 DO
1560     MEM(MEM($2E,Y1),MEM($2F,Y1)+I):=LINE[I]
1570 END
1580 %-----------------------------
1590 LEFT
1600 VAR I,J,L,M
1610 BEGIN
1620   FOR I:=Y1 TO Y2 DO [
1630     L:=MEM($2E,I)
1640     M:=MEM($2F,I)
1650     TEMP:=MEM(L,M+X1)
1660     FOR J:=X1 TO X2-1 DO
1670       MEM(L,M+J):=MEM(L,M+J+1)
1680     MEM(L,M+X2):=TEMP ]
1690 END
1700 %-----------------------------
1710 RIGHT
1720 VAR I,J,K,L,M
1730 BEGIN
1740   FOR I:=Y1 TO Y2 DO [
1750     L:=MEM($2E,I)
```

1

```
1760      M:=MEM($2F,I)
1770      TEMP:=MEM(L,M+X2)
1780      FOR J:=X2 DOWNTO X1+1 DO
1790        MEM(L,M+J):=MEM(L,M+J-1)
1800      MEM(L,M+X1):=TEMP ]
1810 END
```

2.

리스트 15 : HGR화면 스크롤 프로그램

```
1000 PROC
1010   LEFT,RIGHT,UP,DOWN
1020 VAR
1030   Z,I,J,K,L,M,N,O,TEMP
1040 ARRAY
1050   BH[23],BL[23],LINE[39]
1060 BEGIN
1070   FOR I:=0 TO 2 DO
1080     FOR J:=0 TO 7 DO [
1090       BH[I*8+J]:=$4+J/2
1100       K:=ASR(J)
1110       K:=(0.ADC.0)*$80+I*$28
1120       BL[I*8+J]:=K ]
1130   FOR Z:=1 TO 40 DO LEFT
1140   FOR Z:=1 TO 40 DO RIGHT
1150   FOR Z:=1 TO 24 DO UP
1160   FOR Z:=1 TO 24 DO DOWN
1170 END
1180 %-----------------------------
1190 LEFT
1200 BEGIN
1210   FOR I:=0 TO 23 DO [
1220     L:=BH[I]
1230     M:=BL[I]
1240     TEMP:=MEM(L,M)
1250     FOR J:=0 TO 38 DO
1260       MEM(L,M+J):=MEM(L,M+J+1)
1270     MEM(L,M+39):=TEMP ]
1280 END
1290 %-----------------------------
1300 RIGHT
1310 BEGIN
1320   FOR I:=0 TO 23 DO [
1330     L:=BH[I]
1340     M:=BL[I]
1350     TEMP:=MEM(L,M+39)
1360     FOR J:=39 DOWNTO 1 DO
1370       MEM(L,M+J):=MEM(L,M+J-1)
1380     MEM(L,M):=TEMP ]
1390 END
1400 %-----------------------------
1410 UP
1420 BEGIN
1430   FOR I:=0 TO 39 DO
1440     LINE[I]:=MEM(BH[0],BL[0]+I)
1450   FOR I:=0 TO 22 DO [
1460     L:=BH[I]     M:=BL[I]
1470     N:=BH[I+1]   O:=BL[I+1]
1480     FOR J:=0 TO 39 DO
1490       MEM(L,M+J):=MEM(N,O+J) ]
1500   FOR I:=0 TO 39 DO
1510     MEM(BH[23],BL[23]+I):=LINE[I]
1520 END
1530 %-----------------------------
1540 DOWN
1550 BEGIN
1560   FOR I:=0 TO 39 DO
1570     LINE[I]:=MEM(BH[23],BL[23]+I)
1580   FOR I:=23 DOWNTO 1 DO [
1590     L:=BH[I]     M:=BL[I]
1600     N:=BH[I-1]   O:=BL[I-1]
1610     FOR J:=0 TO 39 DO
1620       MEM(L,M+J):=MEM(N,O+J) ]
1630   FOR I:=0 TO 39 DO
1640     MEM(BH[0],BL[0]+I):=LINE[I]
1650 END
```

리스트 16 : TL/1의 라이브러리를 재배치하는 프로그램

```
10  HOME : VTAB 10: HTAB 9: PRINT "TINY LANGUAGE
    / 6502"
20  PRINT "BLOAD TL2,A$9000"
25  VTAB 15: HTAB 6: PRINT "PLEASE SETUP BEGIN A
    DDRESS."
30  PRINT "BLOAD TL1.3,A$8FD"
35  CALL 36864: CALL 2301
```

리스트 17 : 리스트 12를 개선한 TL/1 프로그램

```
1000 %=========================
1010 %
1020 % THIS PROGRAM CONVERTS
1030 % HGR IMAGE TO CHAR PLOT
1040 %
1050 % 1987.9.10  LEE CHAN JIN
1060 %
1070 %-------------------------
1080 %
1090 % TO USE THIS PROGRAM
1100 % FIRSTLY
1110 %   BLOAD PIC.NAME,A$4000
1120 % AND EXECUTE IT.
1130 %
1140 %=========================
1150
1160 PROC
1170   XDRAW,INVERT
1180 VAR
1190   I,J,X1,Y1,X2,Y2,XH,XL,Y,SUM,CHAR
1200 BEGIN
1210   MEM($00,$E6):=$40 % HGR 2 PAGE
1220   MEM($00,$E7):=1   % SCALE=1
1230   MEM($00,$E8):=$00 % SHAPE TABLE
1240   MEM($00,$E9):=$3E
1250   MEM($3E,$00):=1    % DATA
1260   MEM($3E,$01):=0
1270   MEM($3E,$02):=4
1280   MEM($3E,$03):=0
1290   MEM($3E,$04):=53
1300   MEM($3E,$05):=55
1310   MEM($3E,$06):=37
1320   MEM($3E,$07):=0
1330   X1:=MEM($3F,$00)  % PRINT RANGE
1340   Y1:=MEM($3F,$01)
1350   X2:=MEM($3F,$02)
1360   Y2:=MEM($3F,$03)
1370   INVERT            % INVERT HGR
1380   MEM($C0,$50):=0   % DISPLAY HGR
1390   MEM($C0,$57):=0
1400   MEM($C0,$55):=0
1410   MEM($C0,$52):=0
1420   FOR I:=Y1 TO Y2 DO [
1430     Y:=I*3
1440     FOR J:=X1 TO X2 DO [
1450       XL:=J*2
1460       XH:=MHIGH
1470       XDRAW(1,XH,XL,Y)
1480       SUM:=MEM($00,$EA)
1490       CASE SUM OF
1500           6 CHAR:=' '
1510           5 CHAR:='.'
1520           4 CHAR:='-'
1530           3 CHAR:='+'
1540           2 CHAR:='?'
1550           1 CHAR:='*'
1560         ELSE CHAR:='B'
1570         WRITE(0:ASCII(CHAR)) ]
1580       WRITE(0:CRLF) ]
1590 END
1600 %
1610 % XDRAW
1620 %
1630 XDRAW(NO,XH,XL,Y)
```

1

TL/1 언어의 활용

```
1640 VAR
1650   TH,TL
1660 BEGIN
1670   CALL($F7,$30,0,1)
1680   CALL($F4,$11,Y,XL,XH)
1690   TL:=MEM($00,$1A)
1700   TH:=MEM($00,$1B)
1710   CALL($F6,$5D,0,TL,TH)
1720 END
1730 %
1740 % INVERT HGR IMAGE
1750 %
1760 INVERT
1770 VAR
1780   I,J
1790 BEGIN
1800   FOR I:=$40 TO $5F DO
1810     FOR J:=0 TO $FF DO
1820       MEM(I,J):=COM(MEM(I,J))
1830 END
```

리스트 18 : TL/1과 연결되는 베이직 프로그램

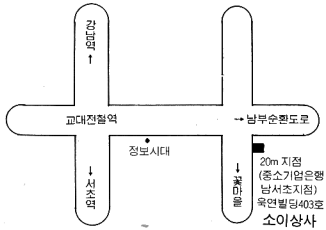
```
10 D$ =  CHR$ (4)
20 E$ =  CHR$ (27)
30  PRINT D$"BLOAD PROGRAM"
40  PRINT D$"BLOAD LIBRARY"
50  PRINT D$"PR#1"
60  PRINT  CHR$ (15);
70  PRINT  CHR$ (27)"M";
80  PRINT E$"3" CHR$ (18);
85  POKE 1657,160
90  PRINT  CHR$ (13);D$"PR#0"
91  POKE 16128,0: POKE 16129,0: POKE 16130,139:
    POKE 16131,63
95 :
100  TEXT : HOME
110  PRINT ":==============:"
120  PRINT "! HGR => CHAR !"
130  PRINT "! ------------ !"
140  PRINT "! 1987 BY LCJ !"
150  PRINT ":==============:"
160  PRINT : PRINT
170  PRINT "1. LOAD NEW PICTURE"
180  PRINT "2. VIEW PICTURE"
190  PRINT "3. CONVERT & PRINT"
200  PRINT "4. CATALOG"
210  PRINT : PRINT
220  PRINT "SELECT ONE : ";
230  GET A$
240  ON  VAL (A$) GOTO 300,340,400,500
250  GOTO 230
260 :
300  PRINT : PRINT : PRINT
310  INPUT "FILE NAME : ";FI$
320  IF FI$ = "" THEN 100
330  PRINT  CHR$ (13) CHR$ (4)"BLOAD "FI$",A$400
     0"
340  GOSUB 600: GOTO 100
400  PRINT : PRINT : PRINT
410  INPUT "SET PRINTER POSITION & <RETURN> ";A$

420  PRINT  CHR$ (4)"PR#1"
430  CALL 37376
440  PRINT  CHR$ (4)"PR#0"
450  GOTO 100
500  HOME
510  PRINT  CHR$ (13) CHR$ (4)"CATALOG"
520  PRINT
530  PRINT "PRESS ANY KEY.";
540  GET A$: GOTO 100
600  POKE  - 16297,0: POKE  - 16304,0: POKE  - 1
     6299,0: POKE  - 16302,0
610  GET A$: RETURN
```

이전안내

금번 폐사는 사세확장에 따라
사무실을 아래와 같이 이전하게 되었읍니다.
앞으로도 많은 성원을 바랍니다.

주소 : 서울시 강남구 서초동 1597-6호
(옥연B/D 403호)
C.P.O. BOX : 531
TLX : K28736 SOYEE
TEL : 587-2431~3(3LINE)
FAX : (02)587-2434



소이상사