



애플 TL/1

작은 컴파일러 언어이지만 거인 컴파일러 언어만큼이나 큰 작업을 시도하려는 TL/1. 일명 꼬마 컴파일러 언어/ 이번 호에서는 TL/1이 공개된 지 오래된 지금도 사용법과 문법을 모르는 독자들을 위해서 필자가 분석한 자료를 정리해 보고자 한다. 이미 다른 컴퓨터 잡지에 MSX용 TL/1은 공개되었으나 애플 사용자를 위해 페이지를 할애하였으므로 TL/1을 사용해 보려는 독자들에게는 좋은 자료가 될 것이다.

TL/1이란?

도대체 TL/1이 무엇일까? 어떤 언어일까? 왜 안대혁이가 저렇게 TL/1을 떠벌일까?

먼저 TL/1에 대해서 알고 싶어하는 독자들을 위해서 TL/1이 무엇인지 설명하기로 하겠다. TL/1이란 일본에서 1980년에 개발된 메모리 상주형 간이 컴파일러 언어이다. TL/1의 TL은 'Tiny Language'의 약자이고, 1은 TL/1 컴파일러가 1패스만에 컴파일을 하기 때문에 붙여진 이름이다.

또한 TL/1은 1바이트 변수를 채용한 컴파일러 언어로서, 컴파일러 언어로서는 가장 간결하고 오브젝트의 효율이 높으며 사용하기 편리한 언어이다. 이

러한 장점이외에도 언어의 형식이 파스칼을 따르고 있어서 파스칼로 작성된 프로그램을 약간의 수정만으로 TL/1으로 옮길 수 있으며, 파스칼처럼 재귀 호출(Recursion)이 되는 장점도 가지고 있다. TL/1과 파스칼의 비교를 위해서 이 글의 뒷부분에서 차이점을 상세히 소개하겠다.

TL/1은 그 이름과 같이 컴파일러 언어로서는 비교적 짧은 길이이므로 다른 기종으로의 이식이 쉬운 편이다. TL/1이 이식된 기종을 살펴 보면 다음과 같다.

- 애플 II 계열
- PC-8001 계열
- MSX 계열
- H68 계열

• MZ 계열

이렇게 주로 6800, 6502, Z80 마이크로프로세서에 이식되어 사용되고 있으므로 국내 기종인 SPC-1000이나 패미콤 기종에도 쉽게 이식할 수 있을라 본다.

이번 호에는 페이지 관계상 TL/1의 덤프 리스트만을 게재하지만 후에 애플 TL/1의 소스 리스트를 발표할 예정이므로 게재된 소스 리스트를 보고 관심 있는 독자들은 다른 기종에 이식해 보기 바란다.

TL/1의 개발목적은 주로 하드웨어 개발이나 게임 프로그램, 그래픽 언어로 쓰여지기 위해서 개발된 것이다. 이유는 TL/1의 수행 속도와 컴파일 속도가 대단히 빠르므로 하드웨어 개발이나 게임 프로그램을 개발할 때 아주 유리하기 때문이다.

지금까지 TL/1의 장점만을 일일이 나열하였는데 이러한 TL/1에도 단점은 있다. 단점은 TL/1에서 사용되는 변수의 길이가 1바이트 이내라는 것이다. 이것은 TL/1의 빠른 속도를 위한 제한 사항이기도 하다. 이렇게 1바이트

TL/1이란?

애플 TL/1의 구조
애플 TL/1 예제

1. N (New)
2. A (Address)
3. L (Load)
4. S (Save)
5. C (Compile)
6. X (eXecute)
7. F (Free)

8. R (Renumber)
9. M (Monitor)
10. O (List)

애플 TL/1 언어의 사양
애플 TL/1 명령어

1. 소스 프로그램의 기본 서식
2. 프로그램의 전체 서식
3. 부프로그램명 선언
4. 주 프로그램
5. 부 프로그램

6. 설명문
7. 변수
8. 식

애플 TL/1 입력 및 저장방법
애플 TL/1 사용예
애플 TL/1과 파스칼의 비교
그밖의 참고사항
맺음말

변수만을 사용함으로써 컴파일러의 부담도 줄어 주고 고속으로 소스 프로그램을 컴파일시킬 수 있다. 만일 텍스트가 303행(약 5280바이트)일 때 컴파일 시간과 리스트 출력시간을 비교해서 표시하면 다음과 같다.

- 컴파일하고 리스트하는 시간 : 15.5초
- 컴파일 리스트 출력시간 : 7.0초
- 오브젝트의 길이 : 3920 바이트

이 출력시간을 보면 알 수 있듯이 303행을 컴파일하는데만 걸린 시간은 8.5초이다. 따라서 한 행을 컴파일 할 때 소요되는 시간은 0.028초라는 적은 수치가 계산된다. 이 속도는 터보 파스칼도 따라오지 못할 속수이다.

또한 위에서 보듯이 다른 컴파일러 언어에 비해서 소스 프로그램과 오브젝트의 길이가 많이 차이 나지 않는다. 얼마나 멋진 컴파일러 언어인가?

애플 TL/1의 구조

원래 TL/1은 H68이라는 컴퓨터(6800 CPU 사용)를 사용하여 작성되었다. 애플 TL/1은 H68 TL/1을 그대로 옮긴 것으로 사용방법은 거의 비슷하나 그 구조에 있어서는 상당한 차이점을 보인다. 또한 애플 TL/1은 H68 TL/1에서는 사용할 수 없었던 몇 가지 명령어들을 추가하여 사용할 수 있도록 수정된 부분도 있다.

그림 1은 애플 TL/1의 구조를 메모리 맵으로 나타낸 것이다. 그림 1을 보면 애플 TL/1의 구조를 이해하는데 도움이 되리라 본다.

그림 1에서 보는 바와같이 컴파일러 본체는 항상 \$8FD번지에 위치하며, 런타임 루틴은 메모리의 상위 번지에 놓여진다. 특히 런타임 루틴은 페이지별로 재배치(Relocatable)가 가능하므로 사용자가 원하는 번지로 옮길 수 있다. 재배치 방법은 뒤에서 다시 설명하겠다.

그림 1 : 애플 TL/1의 메모리 맵(48K 램의 경우)

\$0000	포인터 플래그	
\$0080	광역 변수 영역	
\$0100	하드웨어 스택	
\$0200	입력 버퍼 및	
	컴파일러용 스택	
\$0300	사용자 영역 및	
	도스 엔트리 포인터	
\$0400	텍스트 페이지 1	
\$0800	워크 테이블	
\$08FD	컴파일러 본체	} TL1 (리스트 3)
\$1900	식별자 테이블(Identifier Table)	
\$1C00	에디터	
\$2000	소스 텍스트와	
	오브젝트 영역	} TL2 (리스트 4)
\$7E00	런 타임 루틴	
\$8300	변수 워크 에리어	
\$9600	도스	
\$BFFF		

애플 TL/1 에디터

필자는 TL/1의 에디터가 너무나 불편하여 간이형 라인 / 스크린 에디터를 만들어 사용하고 있는데 이 또한 불편하기 이룰데 없다. 기회가 있다면 다음에 관한 에디터를 작성하여 게재할 생각이다. 일단 여러분은 현재의 시점까지 완성된 에디터를 사용하지 바란다. 다음은 간이 에디터의 사용명령을 나열한 것이다.

1. N(New)

현재 메모리안에 있는 소스 프로그램을 지운다.

2. A(Address)

소스 프로그램의 시작번지를 지정한다. 이때 번지는 16진수로 입력해야 한다.

3. L(Load)

소스 프로그램을 로드한다. L명령을 내리면 'FILE?'이라고 물어 오는데, 이때 화일명을 입력하면 디스켓에서 프로그램을 로드한다. 만일 화일명을 입력시키지 않고 리턴키만 누르면 애플 TL/1의 에디터 모드로 돌아 간다.

4. S(Save)

소스 프로그램을 디스켓에 저장한다. 사용방법은 L명령과 같다.

5. C(Compile)

소스 프로그램을 컴파일시킨다.

6. X(Execute)

오브젝트 프로그램을 실행시킨다.

7. F(Free)

사용중인 소스 프로그램의 메모리 영역을 표시한다.

8. R(Renumber)

메모리에 있는 소스 프로그램을 100행부터 10씩 증가시키면서 행번호를 재배치한다.

9. M(Monitor)

애플의 모니터 모드로 빠져 나온다. 다시 에디터 모드로 가고 싶을 때는 <Ctrl Y>키를 누르고, 리턴키를 누르면 된다.

10. O(List)

소스 프로그램을 리스트한다.

애플 TL/1 언어의 사양

기본적으로 애플 TL/1은 H68 TL/1을 바탕으로 설계되었지만 언어 사양은 조금씩 다르다. 애플 TL/1과 다른 기존의 TL/1과의 차이점을 몇 가지 소개하겠다.

• 애플 TL/1은 IF~THEN~ELSE문 사용가능

애플 TL/1에서는 IF~THEN~ELSE문을 사용할 수 있다. 따라서 다음과 같은 예는 H68 TL/1과는 다르다.

(예) CASE~OF

```

<식1> <문1>
:
<식n> IF~THEN <문n>
ELSE <문n+1>

```

이 예는 H68 TL/1에서는 문제가 없지만 애플 TL/1에서는 에러로 처리된



행시에는 그 초기치가 주프로그램이나 다른 부프로그램 또는 그 부프로그램 자신에 따라서 결정된다. 따라서 지역 변수(Local Variable)라고 말한다. 지역변수는 그 부프로그램에서만 유효하고, 주프로그램이나 다른 부프로그램의 변수와 같은 이름을 사용해도 상관없다. 지역변수는 같은 이름을 사용해도 다른 영역에 설정되어 있으므로 광역 변수나 호출한 프로그램의 변수에 영향을 주지 않는다. 지역 배열 변수의 크기도 광역 변수와 같이 256 바이트이내로 제한되어 있다. 따라서 256바이트내에서 지역변수를 사용해야 한다.

함수 부프로그램의 경우에는 함수값을 돌려 주기 위해서 반드시 실행문 리스트내에 다음과 같은 명령을 내려야 한다.

RETURN 식 (예 : RETURN A*2)

함수 부프로그램과 프로시저어 부프로그램을 사용한 예를 들면 다음과 같다.

(함수 부프로그램)

```
ADD(B,C)
BEGIN RETURN B+C END
```

(프로시저어 부프로그램)

```
SOUND(SCALE,TIME) VAR I,J
BEGIN
  FOR I:=0 TO TIME DO
    FOR J:=0 TO 255 DO
      MEM($03,$00):=MEM
        ($03,$00)+SCALE
    END
```

6. 실행문

6.1 STOP

이 명령은 프로그램의 실행을 정지시킨 후에 에디터로 돌아 오게 하는 역할을 한다. 주프로그램의 끝에는 자동적으로 삽입되는 명령이지만 주프로그램이나 부프로그램내의 임의의 장소에 사

용할 수도 있다.

6.2 RETURN

이 명령은 프로시저어에서 복귀하는 역할을 한다. 이 문은 프로시저어 부프로그램의 끝에 자동적으로 삽입되므로 표시하지 않아도 되지만 프로시저어 안에서 사용할 수도 있다. 단 H68 TL/1에서는 사용할 수 없다.

6.3 RETURN 식

이 문은 식의 값을 함수값으로 하여 함수 부프로그램에서 복귀하는 역할을 한다. 이 명령은 함수 부프로그램에서 반드시 1개이상 사용해야 한다. 이 명령은 컴파일할 때 체크되지 않으므로 주의해서 사용해야 한다. 이 명령은 H68 TL/1에서는 사용할 수 없다.

6.4 FOR (단변수) := <식1> (TO/DO WNTO) <식2> DO <문>

이 명령은 <식1>의 값을 단변수에 대입하고 <식2>의 값을 마지막 값으로 해서 <문>을 실행하는 동작을 한다. 증분은 TO의 경우는 +1, 'DOWNTO'의 경우는 -1로 된다.

H68 TL/1에서는 DOWNTO 기능은 없다. 또한 <문>중에서의 RETURN도 H68 TL/1에서는 사용할 수 없고 <문>이 여러 개일 경우에는 <문>을 확장하여 1개로 합칠 수 있다. TL/1은 문에 대한 확장기호로서 'BEGIN~END', { }, [], ()를 자유롭게 사용한다. 문을 확장했을 경우에는 접속기호가 꼭 필요하지는 않지만 ';'을 접속기호로 사용할 수 있다. 또한 필요하지 않은 문의 확장은 컴파일시에 무시되므로 필요이상으로 많이 사용하지 않는 편이 좋다.

6.5 REPEAT <문 리스트> UNTIL <식>

이 명령은 <식>의 값이 0(거짓)이 되지 않을 때까지 <문 리스트>를 반복수행한다. 무한 루프를 사용할 때는 <문 리스트>가 없고 <식>만 사용할 수 있다.

6.6 WHILE <식> DO <문 리스트>

이 명령은 <식>의 값이 0(거짓)이면 <문 리스트>를 수행하지 않고 그 다음 문장을 수행한다. 식의 값이 0이 아닐 경우에는 <문 리스트>를 실행해서 <식>을 다시 평가한다.

6.7 IF <식> THEN <문 리스트 1> [ELSE <문 리스트 2>]

이 명령은 식의 값이 1(참)이면 <문 리스트 1>을 실행한다. 만일 0(거짓)일 때 ELSE가 있으면 <문 리스트 2>를 실행하고, ELSE가 없으면 다음 행을 실행한다. H68 TL/1에서는 ELSE절은 사용할 수 없다.

6.8 CASE <식0> OF

```
<식1> <문1>
:
:
<식n-1> <문n-1> (n≥2)
ELSE <문>
```

이 명령은 <식 0>의 값을 기준값으로 해서 <식 1>의 값과 비교한다. 만일 같으면 <문 1>을 실행하고 <문 n>의 다음 문장으로 진행한다. 만일 <식 0>가 <식 1>의 값과 같지 않으면 <식 2>의 값과 비교한다. 이때 <식 0>가 <식 2>의 값과 같으면 <문 2>를 수행하고, 같지 않으면 다시 다음 식으로 진행한다. 또한 <식>의 장소에 ELSE문이 있으면 무조건 <문 n>을 수행한다. ELSE는 <식 0>가 <식 1>~<식 n-1>과 같지 않을 때 수행된다. 여기서 ELSE문은 반드시 표시해야 하며, <문>에는 널({})을 기입할 수도 있다. CASE문의 예를 들면 다음과 같다.

```
(예1) IF ~THEN~ELSE 문과 같은 CASE
      E문
CASE 조건식 OF
  TRUE (참값일 때의 처리)
  FALSE (거짓값일 때의 처리)
  ELSE
```

(예2) 널을 사용한 CASE문

```
CASE CHR OF
  'A'   (B:=B+1)
  'N'   {}
ELSE {}
```

여기서 '1개의 문자'는 아스키 문자 정수를 나타낸다.

(예3) 여러 개의 조건식에 의한 분기

```
CASE TRUE OF
  조건식1 {문 1}
  :
  조건식n-1 {문 n-1}
ELSE {문n}
```

6.9 WRITE (식 : 출력 리스트)

식의 값으로 지정된 출력장치에 출력 리스트의 내용을 출력한다.

6.9-1 식

식의 값으로 출력장치를 표시한다.

6.9-2 # (식 1, 식 2)

식 1의 값을 탭 (Tab) 으로 해서 식 2의 값을 오른쪽으로 붙여 출력한다.

6.9-3 "문자열"

" "로 둘러 싸인 문자열을 출력한다.

6.9-4 ASCII (식)

식의 값을 아스키 코드로 출력한다.

6.9-5 SPACE (식)

식의 값만큼 공백을 출력한다. 식의 값이 0일 경우에는 아무 것도 출력되지 않는다.

6.9-6 CRLF

다음 행으로 넘어 간다. (CR+LF입)

6.9-7 CRLF (식)

식의 값만큼 행을 바꾼다. 식의 값이 0일 경우에는 행을 바꾸지 않는다.

6.9-8 HEX (식)

식의 값을 16진수로 출력한다. 이 명령은 H68 TL/1에서는 사용할 수 없다.

6.10 변수 : =식

변수, 변수, ..., 변수 = 식

이 문은 대입문으로서 식의 값을 변수에 전달하는 역할을 한다. 왼쪽에 여러 개의 변수가 있을 경우에는 모든 변수에 오른쪽에 있는 식의 값이 대입된다.

6.11. 프로시듀어의 이름

프로시듀어의 이름 (식, 식, ..., 식)

애플 TL/1 명령어 찾아 보기(알파벳순)

ADC	8.3-5	IF~THEN	6.7	ROR(식)	8.2-13
AND	8.3-4	INP	7.4	RRC(식)	8.2-17
ARRAY	4	LSR(식)	8.2-10	SBC	8.3-5
ASCII(식)	6.9-4	LT	8.3-3	SENSE	6.13
ASCII(정수)	8.1-3	MEM 변수	7.3	SPACE(식)	6.9-5
ASL(식)	8.2-12	MHIGH	8.2-2	STOP	6.1
ASR(식)	8.2-11	MOD	8.2-3	THEN	6.7
BEGIN~END	6.4	NEG(식)	8.2-8	TRUE	8.1-4
CALL	6.12	NOT(식)	8.2-7	UNTIL	6.5
CASE~OF~ELSE	6.8	OF	6.8	USR(...)	8.2-15
COM(식)	8.2-9	OR	8.3-4	VAR	4
CRLF	6.9-6	OUT	7.4	WHILE~DO	6.6
CRLF(식)	6.9-7	PORT(식)	7.4	WRITE(식 : 출력 리스트)	6.9
DOWNT0	6.4	PROC	3	#	8.3-3
ELSE	6.7, 6.8	RDHEX(식)	8.2-16	#(식1, 식2)	6.9-2
EOR	8.3-4	READ(식)	8.2-6	=	8.3-3
FALSE	8.1-4	REPEAT~UNTIL	6.5	<	8.3-3
FOR~TO~DO	6.4	RETURN	6.2	>	8.3-3
FUNC	3	RETURN(식)	5, 6.3	"문자열"	6.9-3
GET(식)	8.2-5	RLC(식)	8.2-18		
GT	8.3-3	RND(식)	8.2-4		
HEX(식)	6.9-8	ROL(식)	8.2-14		



다. 그 이유는 CASE문의 제일 끝에 있는 ELSE문이 ELSE문 바로 직전의 IF에 속하는 ELSE문으로 인식되기 때문이다. 따라서 에러를 막기 위해서는 (식n) 다음을 BEGIN~END문이나 그 문에 해당하는 특수문자 즉, [], (), { }로 묶어 줄 필요가 있다. 이 점만 주의하면 별다른 에러는 발생되지 않는다.

• RETURN문을 FOR문내에 사용가능

애플 TL/1은 RETURN문을 FOR문내에 사용할 수 있다.

• FOR~DOWNT~DO문이 있음

이 문은 -1의 루프를 사용할 수 있도록 할 수 있다.

• Hex (e)라는 서식 사용가능

애플 TL/1은 WRITE문의 출력에 많аж에 따라 Hex(e)라고 하는 서식을 사용하여 16진수로 숫자를 출력할 수 있도록 하고 있다.

• RDHEX(e)라는 함수(e는 입력장치 이름)에서 16진수의 입력가능

• 단산 연산자로서 '-'가 있음

• CALL이라는 문이 있음

애플 TL/1은 다음과 같은 4종류의 CALL문을 허용한다.

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

이 문장중 AH와 AL은 번지(Address)의 상위 바이트와 하위 바이트를 말하며, A, X, Y에는 각각 대응되는 6502 레지스터 값이 주어져서 AH와 AL이 가르키는 번지로 점프한다.

• USR함수

애플 TL/1은 USR이라는 함수가 있으며 서식은 CALL문과 같다. 단지 CALL은 문(Statement)이지만 USR은 함수이므로 기계어 서브루틴에서 돌아올때는 어큐뮬레이터의 값을 함수값으로 가지고 돌아 온다.

• 캐리가 반대로 발생

6502는 6800, Z80과는 달리 캐셀을 한 후에 캐리(Carry)가 반대로 발생한다. 따라서 캐셀을 하고 나서 캐리 플래그를 사용할 경우에는 주의해야 한다.

애플 TL/1 컴파일러는 오브젝트를 소스 프로그램의 뒤에 생성하며, 변수의 시작번지를 항상 런 타임 루틴의 뒤에 위치시킨다. 또한 H68 TL/1에서는 광역 변수의 갯수를 배열 변수를 합하여 256개까지 선언할 수 있지만 애플 TL/1에서는 광역 변수를 최대 128개까지만 사용할 수 있다. 이러한 광역 변수는 제로 페이지의 \$80부터 \$FF까지 사용한다. 이것은 실행 속도를 향상시키기 위해서 배려된 사항이다. 이러한 광역 변수의 사용 수칙에도 불구하고 광역 배열 변수는 최대 256개까지 선언이 가능하다. 그 이유는 광역 변수를 제로 페이지가 아니라 변수의 시작번지로 옮겨서 마치 프로시저어를 선언하여 생성된 것과 같이 사용할 수 있기 때문이다.

이러한 사항이외에도 애플 TL/1의

에디터를 동작시켰을 때 주의할 사항은 L 명령을 사용할 때이다. 만일 N 명령을 내리지 않은 상태에서 L 명령을 사용하여 프로그램을 로드하면 로드된 프로그램이 메모리에 있는 소스 프로그램의 뒤에 붙여진다. 따라서 자주 사용되는 프로시저어는 한 개의 화일에 넣은 후에 다른 프로그램과 어펜드(Append)시켜서 라이브러리처럼 사용하면 편리하다.

표 1은 TL/1을 사용할 때 발생할 수 있는 에러 메시지를 설명한 것이다. 또한 표 2와 3은 지금까지 설명한 내용을 요약한 것이므로 참고하기 바란다.

애플 TL/1 명령어

1. 소스 프로그램의 기본 서식

애플 TL/1의 소스 프로그램은 기본적으로 행이라는 개념이 있는 텍스트로 표현되어 있으면 된다. 따라서 각각의 행과 각 문의 함수는 베이직과는 달리 특별한 제약은 없다. 그러므로 한

표 1 : 애플 TL/1의 에러 메시지

에러 메시지	설 명
Too Many Procs	프로그램내에서 정의된 함수와 프로시저어가 합해서 85개를 초과했다.
Invalid Return	메인 프로그램안에 RETURN문이 있다.
Simple var Expected	FOR루프의 제어변수가 단변수가 아니다.
Constant Expected	CALL, USR문의 번지가 정수가 아니다.
Too Many Params	CALL, USR문의 파라미터가 3개보다 많다.
Too Many Args	프로시저어를 불러 냈을 때의 증가치가 오버플로우했다.
Unexpected Ident	사용한 키워드가 존재하지 않는다. 즉, 선택스 에러를 의미한다.
Illegal Factor	식중에 포함할 수 없는 사항이 있을 때 발생한다.
Undcled Ident	선언되지 않은 식별자(Identifier)를 사용했다.
Def'd Twice	함수와 프로시저어를 2중으로 정의했다.
Undef	선언되지 않은 함수나 프로시저어를 사용했다.
Err div 0	0으로 나누었다.
Too Many Identifier	식별자 테이들이 오버플로우했다.
Object Too Long	오브젝트 프로그램의 길이가 너무 길다. 오브젝트의 길이가 \$8000을 넘었을 경우에 발생하나 거의 발생 가능성이 없다.

행에 여러 개의 문을 사용해도 되고, 한 개의 문을 여러 개의 행으로 나눠서 표시해도 상관없다. 다만 여러 개의 행에 나눠서 표시할 수 없는 최소단위가 있다. 그것은 스테이트먼트(Statement)이다. 이 스테이트먼트는 그 이상 분리할 수 없는 것으로서 분리하면 그 의미가 변할 가능성이 있는 것이다. 스테이트먼트에 해당하는 것은 선언이나 명령의 키워드인 PROC, FOR, DO 등과 접속용 기호, 함수, 변수의 이름, 정수 연산자 등이 있다. 단 대입기호인 '='는 2개의 스테이트먼트로 취급된다.

또한 주의할 필요가 있는 것은 WRI TE문중에 사용되는 문자열 출력은 " "으로 둘러 쌓인 문자열이 한 개의 스테이트먼트로 취급된다는 것이다.

어떤 스테이트먼트에 따라서는 영·숫자로 이루어진 짧은 스테이트먼트를 연속해서 사용할 수 없다. 예를 들어 다음과 같은 문장은 에러를 발생한다.

```
FOR I:=K TO 100 DO
```

에러발생 요인은 이 문장이 'FOR'와 'K TO 100 DO'라는 문자로 임해지기 때문이다. 따라서 다음과 같이 공백을 삽입하여 분리할 필요가 있다.

```
FOR I:=K TO 100 DO
```

스테이트먼트와 스테이트먼트 사이에 삽입해도 에러가 발생되지 않는 문자로는 '\$IF'이하의 코드(주로 행의 끝)', '공백(스페이스)', '.', ',', ';'가 있다. 이 문자들을 잘 이용하면 보기 쉬운 프로그램을 만들 수 있다.

2. 프로그램의 전체 서식

(형식)
부프로그램명 (Procedure) 선언
주프로그램 (Main Program)
부프로그램 리스트
/ 끝 표시 (소스 입력루틴에서 작성)

3. 부프로그램명 선언부

(형식)
PROC 프로시저어 리스트
FUNC 함수명 리스트

부프로그램명은 반드시 PROC를 선언한 다음에 FUNC를 선언해야 한다. 프로시저어 부프로그램이나 함수 부프로그램이 없을 때에는 이 선언을 생략한다.

프로시저어와 함수 부프로그램의 명칭은 첫번째 문자가 영문자이어야 한다. 두번째 문자부터는 영·숫자로서 길이는 자유이다.

부프로그램명의 리스트가 여러 개일 경우에는 ', '로 구분한다. 예를 들어 부프로그램명을 선언하는 예는 다음과 같다.

예)

```
PROC CAR, SCORE,
INITIALIZE,
ST1, ST2
FUNC STICK, LOCATION
```

4. 주프로그램

(형식)
VAR : 광역 변수 선언
ARRAY : 광역 배열과 크기 선언
BEGIN
실행문의 리스트
END

이 형식에서 광역 변수란 주프로그램에서 사용된 변수를 부프로그램에서도 같은 이름으로 읽고 쓸 수 있는 변수를 말한다. 여기서 변수명은 영문자로 시작하는 임의의 영·숫자를 말한다.

배열을 선언할 때는 '[']'을 사용한 다. 배열을 선언하고 난 후에 사용할 수 있는 첨자의 범위는 0에서부터 선언한 값까지이다. 주프로그램의 예를 보면 다음과 같다.

```
VAR LOOP, DUMMY
ARRAY D[10], E[10]
BEGIN
FOR LOOP:=0 TO 10 DO (
D[LOOP], E[LOOP],
DUMMY:=RND(100)
WRITE(0: CRLF,
LOOP,
#(4, DUMMY)) )
END
```

5. 부프로그램

(형식)
부프로그램명 (가인수 리스트)
VAR (지역 변수명 선언)
ARRAY (지역 배열명, 크기 선언)
BEGIN
실행문 리스트
END

부프로그램인 프로시저어와 함수의 이름은 부프로그램명 선언부에서 구분해 주어야 한다. 만일 인수를 가지지 않은 부프로그램이라면 가인수 리스트는 생략해야 한다.

가인수는 Call by Reference 만으로 한정되어 있으므로 부프로그램의 실

표 2: 애플 TL/1의 라이브러리 기능

문	SYS(AH, AL [, A [, H [, L]]])
함수	MUL16 (AH, AL, BH, BL) DIV16 (AH, AL, BH, BL) MODH

표 3: 애플 TL/1의 확장기능

구문	IF~THEN~ELSE
문	CALL (AH, AL [, A [, H [, L]]]) SENSE
함수	USR (AH, AL [, A [, H [, L]]]) RDHEX (e)
기타	·WRITE문중 출력 지점자 HEX (e) ·단행 연신자 '~' ·FOR~DOWNTOW~DO구문의 사용가능 ·FOR 루프내에서 RETURN문 사용가능



이것은 프로시저어 부프로그램을 호출하는 문이다. 인수가 있는 프로시저어에서는 실제의 값(실인수)을 '실인수 리스트'로 지정한다. 여기서는 Call by Reference만으로 한정되어 있으므로 실인수가 변수일지라도 프로시저어에서 돌아온 값은 변하지 않는다. 단, 광역 변수로 취급하고나서 변경하면 값은 변할 수 있다.

6.12 CALL(AH, AL [, A [, H [, L]]])

이 문은 기계어 서브루틴을 호출하는 명령이다. 각 파라미터의 값은 다음과 같다.

- AH: 번지의 상위 바이트
- AL: 번지의 하위 바이트
- A: 어큐뮬레이터에 주는 값
- H: X레지스터에 주는 값(PC-8001 계열은 H레지스터)
- L: Y레지스터에 주는 값(PC-8001 계열은 L레지스터)

이 명령을 내릴 때는 A, H, L 레지스터는 없어도 상관없다. 이 명령은 애플 TL/1과 PC-8001계열 TL/1에서만 유효하다.

6.13 SENSE

이 명령은 STOP 키(PC-8001의 경우)나 <Ctrl C>(애플의 경우)가 눌러졌는가를 체크하고, 눌러져 있으면 에디터로 돌아오는 역할을 한다.

7. 변수

애플 TL/1에서 사용되는 변수의 길이는 1바이트이다. 또한 변수에는 다음과 같은 종류가 있다.

7.1 단변수

VAR로 선언되어 있고, 영문자로 시작하는 변수를 말한다. 즉, 모든 광역 변수와 지역 변수가 이에 해당된다.

7.2 배열 변수

이 변수는 '배열 변수명[식]'의 형태

로 나타내며, ARRAY로 선언된 변수를 말한다.

7.3 MEM 변수

'MEM(식1, 식2)'의 형태로 나타내며 식1의 값을 상위 번지, 식2의 값을 하위 번지로 하는 메모리내의 1바이트 변수를 말한다.

7.4 PORT(식)

Z80계열 전용의 명령어이다. 이 명령은 N비트의 INP, OUT에 대응하고 있다. 불행하게도 이 명령은 애플 TL/1에는 존재하지 않는다.

8. 식

8.1 점수

8.1-1 10진 점수
0~255까지의 10진 점수이다.

8.1-2 16진 점수
\$00~\$FF까지의 16진 점수이다.

8.1-3 문자 점수
아스키 문자로 된 한 문자의 점수이다.

<형식>
ASCII "한 문자"

8.1-4 논리점수

TRUE(=\$FF)
FALSE(=\$00)

8.2 함수

8.2-1 함수 부프로그램명

<형식>
함수 부프로그램명 (식1, ..., 식N)
실인수 리스트

함수값은 프로그램안에서 정의된 함수 부프로그램에 의해서 결정된다. 인수가 있을 경우에는 '(실인수 리스트)'에 의해서 함수값이 정해진다.

8.2-2 MHIGH

곱셈을 계산할 때의 계산결과를 2바이트로 얻어지지만 하위의 1바이트만 표현된다. 이 MHIGH함수는 상위와 1바이트 값을 취하는 시스템 변수이다. 이 함수는 문법상 대입할 수 없으므로 함수로서만 취급된다.

8.2-3 MOD

나눗셈을 계산할 때 나머지가 들어 있는 시스템 변수이다. 대입할 수는 없다.

8.2-4 RND(식)

이 함수는 '1≤RND(식)≤식' 사이의 정수형 난수를 발생시키는 역할을 한다.

8.2-5 GET(식)

식으로 주어진 값의 입력장치에서 한 문자를 입력받고 그 문자의 아스키 코드값을 얻는 역할을 한다.

8.2-6 READ(식)

식으로 주어진 값의 입력장치에서 10진수를 한 개 입력받고 입력받은 값을 얻는다.

8.2-7 NOT(식)

식의 값을 1의 보수로 취하는 함수이다.

8.2-8 NEG(식)

식의 값을 2의 보수로 취하는 함수이다.

8.2-9 COM(식)

식의 값을 1의 보수로 취한다. NOT와 같다.

*8.2-10 LSR(식)

식의 값을 오른쪽으로 1비트 쉬프트한다. MSB에는 0이 입력되고, LSB

주) *가 붙은 함수는 6502의 기계어와 같은 기능을 한다.

의 값은 캐리(Carry)로 들어 간다.

8.2-11 ASR(식)

식의 값을 오른쪽으로 1비트 쉬프트한다. MSB는 변하지 않고 LSB의 값이 캐리로 들어 간다.

*8.2-12 ASL(식)

식의 값을 왼쪽으로 1비트 쉬프트한다. MSB는 캐리로 입력되고, LSB에는 0이 들어 간다.

*8.2-13 ROR(식)

식의 값을 오른쪽으로 1비트 로테이트(Rotate)시킨다. LSB에는 캐리값이 들어 가고, MSB의 값은 캐리로 들어 간다.

*8.2-14 ROL(식)

식의 값을 왼쪽으로 1비트 로테이트시킨다. MSB에는 캐리값이 들어 가고, LSB의 값은 캐리로 입력된다.

예) ROL, ROR의 사용예(비트 순서의 역전 함수)

```
INBIT(A) VAR B,K
BEGIN
  FOR K:=1 TO 8 DO (
    A:=ROR(A)
    B:=ROL(B) )
  RETURN B
END
```

8.2-15 USR(AH, AL [, A [, H [, L]]])

이 문은 CALL문과 기능은 같다. 하지만 함수값으로 기계어 서브루틴 실행 후에 어큐뮬레이터의 값을 가지고 돌아온다. 이 명령은 PC-8001과 애플 TL/1에서만 유효하다.

8.2-16 RDHEX(식)

식으로 주어진 값을 입력장치로 해서 한 항을 입력받는다. 이 명령은 PC-8001과 애플에서만 유효하다.

8.2-17 RRC(식)

PC-8001 전용 캐리를 경유하지 않고 식의 값을 오른쪽으로 1비트 쉬프트한다. 즉, LSB의 값이 MSB로 옮겨 진다.

8.2-18 RLC(식)

PC-8001 전용 캐리를 경유하지 않고 식의 값을 왼쪽으로 1비트 쉬프트한다. 즉, MSB의 값이 LSB로 옮겨 진다.

8.3 2항 연산자

8.3-1 승제산 연산자

*: 승산
/: 제산

8.3-2 가감산 연산자

+: 가산
-: 감산

8.3-3 관계 연산자

관계 연산자는 연산 결과가 TRUE (\$FF) 나 FALSE(\$00)이다.

>: 크다.
<: 작다.
#: 다르다(같지 않다).
=: 같다.
*GT: 크다(Greater Than)
*LT: 작다(Less Than)

이 관계 연산자중 *가 붙은 연산자는 2의 보수로 표현된 부호있는 2진수를 취해서 비교하는 연산자이다. *가 붙지 않은 연산자는 부호없는 2진수를 취해서 비교하는 연산자이다.

8.3-4 논리 연산자

*AND: 논리곱
*OR: 논리합
*EOR: 배타적 논리합

이 연산자들은 2의 보수로 표현된 부호가 붙은 2진수로 비교를 하는 연산자이다.

8.3-5 캐리부 가감산 연산자

*ADC: 캐리부 가산

*SBC: 캐리부 감산(캐리를 부호로) 어떤 컴퓨터 사용자들은 영문자 연산자에 따라서는 '.AND.'와 같이 양쪽에 '(.점)'을 붙이는 것을 표준이라고 말한다. 하지만 없어도 상관없다.

ADC, SBC 연산은 1바이트 이상의 긴 계산에 사용할 수 있지만 컴파일 결과를 생각하지 않으면 사용하기 곤란하다.

예) 2바이트 가산에

```
DADD (C,D,E,F)
BEGIN
  B:=D+F
  A:=C.ADC.E
END
```

이 예는 '(C,D)+(E,F)'를 계산하고 계산값을 광역 변수 (A,B)에 주는 프로그램이다.

애플 TL/1 입력 및 저장방법

애플 TL/1을 사용하기 위해서는 리스트 3과 리스트 4의 덤프 리스트를 입력해야 한다. 모두 입력시켰다면 수정 작업을 끝낸 후에 다음과 같이 디스켓에 저장시킨다.

리스트 3
BSAVE TL1, A,\$8FD, L\$1703 (RETURN)

리스트 4
BSAVE TL2, A,\$7E00, L\$0500 (RETURN)

그리고 리스트 1을 입력시킨 후에 'HELLO'라는 화일명으로 TL1과 TL2가 저장되어 있는 디스켓에 다음과 같이 리스트 1을 새어보시킨다.

SAVE HELLO (RETURN)

이제 애플 TL/1 디스켓은 완성된 셈이다. 완성된 애플 TL/1 디스켓을 부팅시키면 화면에는 그림 2와 같이 표



시된다. 화면의 하단에 나타난 ':' 은 애플 TL/1의 프롬프트(Prompt)이다. 이 상태에서 TL/1을 사용할 수 있다.

먼저 ':' 상태에서 A명령을 내려 소스 프로그램의 시작번지를 지정한다. 보통 소스 프로그램의 시작번지로는 \$2000번지를 지정한다. 번지는 다음과 같이 16진수로 입력하면 된다.

```
:A (RETURN)
ADR$ $2000
:
```

시작번지를 지정하였으면 N키를 눌러 전에 있던 프로그램을 지운다. 'NEW?' 라고 물으면 (Y)키를 누른다. 그러면 메모리내의 소스 프로그램이 지워진다. 이 상태에서 TL/1의 소스 프로그램을 입력하면 된다. N명령을 내렸을 때 화면에 표시되는 메시지는 다음과 같다.

```
:N (RETURN)
NEW ?Y ←Y를 입력한다.
:
```

애플 TL/1 사용예

이제 애플 TL/1으로 프로그램을 작성해 보자. N명령을 내린 상태에서 리스트 2를 입력해 보자. 에디팅 방법은 애플 베이직 모드에서와 같다.

프로그램을 모두 입력했다면 S키를 눌러 프로그램을 디스켓에 저장시킨다. S키를 누르면 화면의 카탈로그가 표시되고 'FILE?' 이라고 묻는데, 이때 파일명을 입력하면 입력한 파일명으로 프로그램을 저장시킨다. 만일 리턴키만 누르면 TL/1 에디터 모드로 빠져 나온다. 파일을 로드하는 L명령도 S명령과 같은 방식이다. 그림 3은 L명령을 사용하여 프로그램을 디스켓에서 로드하는 과정을 보이고 있다.

이제 리스트 2를 컴파일시켜 보자.

리스트 1: 애플 TL/1의 HELLO 프로그램

```
10 TEXT : HOME : A$ = CHR$ (124)
15 PRINT "APPLE DOS 3.3                                81/01
   /28";
20 PRINT "+-----+
   --+";
25 PRINT A$;; INVERSE : PRINT " :
                                   :";: NORMAL : PRINT A$;
   A$;
30 INVERSE : PRINT " : TL/1:APPLE MOD/BY AHN
   DAEHYUK. : ";: NORMAL : PRINT A$;A$;; INVERSE
35 PRINT " :
   ";
40 NORMAL : PRINT A$;"+-----+
   -----+";
45 POKE 34,6
50 PRINT CHR$ (4);"BLOAD TL2,A$7E00"
51 VTAB 10: PRINT "PLEASE SETUP BEGIN ADDRESS."

52 VTAB 20: PRINT "EDITOR>"
60 PRINT CHR$ (4);"BRUN TL1,A$8FD"
70 END
```

그림 2: 애플 TL/1의 초기화면

```
APPLE DOS 3.3                                81/01/28
+-----+-----+
I : I
I : I
I : TL/1:APPLE MOD/BY AHN DAEHYUK : I
I : I
I : I
+-----+-----+

PLEASE SETUP BEGIN ADDRESS.

EDITOR>

:
```

리스트 2: 애플 TL1 언어로 작성한 예제 프로그램

```
10 MAIN PROGRAM
20 VAR I
30 BEGIN
40 FOR I:=0 TO 255 DO
50 WRITE (0:ASCII(I)," ")
60 END
```

그림 3: L명령 사용예

```
:L

DISK VOLUME 254
457 SECTORS FREE

A 003 HELLO
B 025 TL1
B 007 TL2
B 002 TEST
B 002 TEST2
FILE ?TEST2

:
```

C명령을 내리면 컴파일 이 된다. 만일 에러가 발생하면 0키를 누른 후에 리스트를 보고 고칠곳을 찾아 수정하면 된다. 이때 리스트의 출력을 중단시키려면 <Ctrl C>키를 입력하면 되고, 어떤 행을 지우고 싶을 경우에는 지우려는 행의 행번호를 입력하고 리턴키를 누르면 된다. 그림 4는 리스트 2를 컴파일시키는 과정을 나타내었다.

컴파일을 시킨 후에 에러가 없다면 X명령으로 프로그램을 실행시킨다. 만일 행번호를 재배치 (Renumber) 시키려면 R명령을 내리면 된다. 그러면 행번호가 1000부터 10행의 간격으로 바뀌게 된다. 그림 5는 행번호를 재배치시키는 R명령의 실행예를 보이고 있다.

TL/1 프로그램을 로드할 때 주의할 사항은 전에 있던 프로그램을 지우지 않은 상태에서 다른 프로그램을 로드하면 나중에 로드된 프로그램이 전에

그림 4: 컴파일 과정

```
:0

10 % MAIN PROGRAM
20 VAR I
30 BEGIN
40 FOR I:=0 TO 255 DO
50 WRITE (0:ASCII(I)," ")
60 END

:C

%% TL/65 FOR APPLE VER 2.0 BY A.D.H. %%

% MAIN PROGRAM
VAR I
BEGIN
FOR I:=0 TO 255 DO
WRITE (0:ASCII(I)," ")
END

COMPILATION COMPLETE.

OBJECT RUNS AT $8000
OBJECT BEGINS AT $205F
OBJECT ENDS AT $2086
VARIABLE BASE AT $8300

:
```

있던 프로그램에 덧붙여 진다는 점이다. 이 기능은 유용한 프로그램을 여러 개 작성하여 세이브시켜 놓고 합칠 경우 편리하다.

애플 TL/1의 에디터 상태에서 애플의 모니터 모드로 들어 가기 위해서는 M명령을 내려야 한다. 반대로 모니터 상태에서 에디터로 가기 위해서는 '1C 03G' 명령을 내리면 된다. '1C03G' 명령 이외에도 리셋키를 누르거나 <Ctrl Y>키를 입력해도 같은 결과를 얻는다.

프로그램을 프린터로 출력하고 싶으면 그림 6과 같이 M명령으로 모니터 상태로 간 후에 <Ctrl C>키를 눌러 베이직 상태로 가서 PR# 1명령을 내린다. 그리고 다시 CALL-151명령으로 모니터 모드로 온 다음에 <Ctrl Y>키를 눌러 에디터 상태로 온 후에 0명령으로 리스트를 출력하면 된다.

그림 5: R명령 사용예

```
:0

10 MAIN PROGRAM
20 VAR I
30 BEGIN
40 FOR I:=0 TO 255 DO
50 WRITE (0:ASCII(I)," ")
60 END

:R

:0

1000 MAIN PROGRAM
1010 VAR I
1020 BEGIN
1030 FOR I:=0 TO 255 DO
1040 WRITE (0:ASCII(I)," ")
1050 END

:
```



그림 6 : 프린터 사용방법

:M

* ← (Ctrl C)키를 누른다.

JPR#1

ICALL-151

* ← (Ctrl Y)키를 누른다.

:

애플 TL/1 과 파스칼의 비교

애플 TL/1과 파스칼 프로그램을 비교하는 일은 재미있는 작업이다. 표 4는 TL/1과 파스칼 프로그램을 비교 설명해 놓았다. 많은 참고 바란다.

그밖의 참고사항

1. 애플 TL/1의 변수 워크 에어리어를 변경시키기 위해서는 리스트 3에 있는 TL/1의 덤프 리스트중에서 \$980번지에 하위 번지들, \$982에 상위 번지를 입력하면 된다.
2. 애플 TL/1의 런 타임 루틴을 입의 번지로 옮기려면 리스트 3의 TL/1 프로그램을 로드한 후에 리스트 4의 TL/1 프로그램을 'xx00'번지(즉, 1페이지 단위로)로 BRUN시키면 된다.
3. 애플 TL/1에서는 16진수를 사용할 때 숫자 앞에 \$를 입력하면 된다.
(예) X:=\$FF와 X:=-255는 같다.
4. 리넘버 기능은 로드명령의 실행후에는 반드시 수행된다.
5. 48K 시스템에서 정상적인 DOS를 사용하지 않는다면 다음을 입력하여 도스의 엔트리를 바꾸어야 한다.

*1C28:EA EA EA

*1C2D:EA EA EA

6. 애플 TL/1의 \$1C00번지는 에디터

표 4 : 애플 TL/1과 파스칼의 비교

애플 TL/1	파 스 칼	설 명
PROC CAR, SET CAR BEGIN : SET END SET BEGIN : CAR END	PROCEDURE SET; FORWARD; PROCEDURE CAR; BEGIN : SET; END; PROCEDURE SET; BEGIN : CAR; END;	양쪽의 수속절차를 비교한 예이다. 파스칼에서는 FORWARD를 미리 선언해야 하지만 TL/1에서는 선언할 필요가 없다.
CASE X<Y OF TRUE (A:=1) FALSE (A:=0) 또는 IF X<Y THEN A:=1 ELSE A:=0	IF X<Y THEN A:=1 ELSE A:=0	TL/1에서는 CASE문에 식을 사용한 후에 TRUE나 ELSE문으로 바꾸어 처리할 수 있다.
CASE ABC OF 0 (X:=1) 1 (X:=2) 2 (X:=3) ELSE (X:=10)	IF ABC=0 THEN X:=1 ELSE IF ABC=1 THEN X:=2 ELSE IF ABC=2 THEN X:=3 ELSE X:=10;	TL/1에서는 IF~THEN ELSE문 대신에 CASE문을 사용할 수 있다.
CASE ABC OF 0 (X:=1) 1 (X:=2) 2 (X:=3) ELSE (X:=10)	CASE ABC OF 0: X:=1; 1: X:=2; 2: X:=3; 3,9 X:=10;	CASE문의 서식이 약간 다르다.
CASE APPLE=MSX OF TRUE (IF XY=Z THEN PC:=0) ELSE (IF XY=A THEN PC:=255)	IF APPLE=MSX THEN IF XY=Z THEN PC:=0 ELSE BEGIN END ELSE IF XY=A THEN PC:=255	파스칼의 'ELSE IF'와의 차이점을 보여 주고 있다.

의 콜드 스타트(Cold Start)이고, \$1C03번지는 웜 스타트(Warm Start)이다. 따라서 모니터 모드에서 (Ctrl Y)로 에디터로 돌아 올 경우에는 웜 스타트를 실행시키는 것이다.

맺음말

이상으로 애플 TL/1에 대한 간략한 소개를 마친다. 마이크로소프트웨어로 찾아 드는 독자들의 열화와 같은 성화

에 보답하고자 마련한 TL/1 언어가 여러분에게 얼마만큼이나 다가갔는지 궁금해 진다. 필자는 TL/1 언어로 작성된 프로그램이 널리 사용되기를 바라며, 작지만 커다란 힘을 발휘하는 꼬마 TL/1 (Tiny Language)의 많은 보급 또한 바라는 바이다. 또한 기회가 닿는대로 필자가 작성한 TL/1의 소스 리스트를 공개하고자 하니 독자들도 TL/1을 분석해서 나름대로의 컴파일러를 내놓을 설계해 보시는 것이 어떠실런지.

리스트 3: TL1의 덤프 리스트

```
P1: +0 +1 +2 +3 +4 +5 +6 +7 SUM
88Fd- 4C 00 1C 4B
0900- 4C 06 09 4C 00 08 48 A9 48
0908- 00 05 16 05 71 85 65 A5 100
0910- 72 05 68 85 73 85 79 05 140
0918- 78 05 01 AA 74 08 08 CA 197
0920- 48 F2 06 77 A9 28 05 1F d
0928- 74 A5 68 A6 61 18 69 42 103
0930- 90 01 E8 05 7C 86 74 A9 126
0938- 44 A2 19 85 6C 86 64 A4 1A7
0940- 62 A6 63 C8 08 01 E0 8A 170
0948- 76 86 F7 8C 0A 80 0E 0b 132
0950- 08 28 98 16 06 25 28 0b 16C
0958- 54 AC 2F 36 35 28 44 AF 1EF
0968- 52 28 41 50 58 4C 45 28 1A4
0968- 56 45 52 20 32 2E 38 28 1bd
0970- 42 59 28 41 2E 44 2E 40 1E4
0978- 2E 20 25 25 8d 08 0A 15A
0980- 08 A2 83 85 74 86 70 28 145
0988- 1E 8A 28 06 13 C9 38 28 1dF
0990- 85 28 2E 08 68 67 20 1E2
```

SUM 5d 19 9E F5 77 7A 80 AA 124

```
P12: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0990- 0A C9 31 d8 85 28 2E 0A 131
0998- 08 F7 8A 68 C8 56 98 17 1b5
09A0- 28 05 16 05 87 54 AF 12b
09B0- 20 4d 41 AE 59 28 08 52 117
09B8- 4F 43 53 00 4C 69 15 20 1CF
09C0- 1E 0A 42 80 86 6b 20 99 1FA
09C8- 0A 20 F1 0A A5 6C A6 6d 149
09D0- 08 05 67 86 76 68 E8 0F 117
09D8- F8 41 E8 0A 15 5d CA 86 172
09E0- 72 A5 6A 28 CA 15 28 1F 1b5
09E8- 15 03 28 0C 08 28 1E 0A 10C
09F0- 8A 6b 28 0b 13 C9 37 28 1dF
09F8- 08 28 2E 0A A9 3b 20 0b 1EF
0A00- 13 28 99 0A A6 72 88 08 1CF
0A08- 08 03 28 1C 0b 67 A6 14A
0A10- 78 05 4C 8A 6d A2 85 86 181
0A18- 77 08 59 4C 0b 1A 08 08 18d
0A20- 48 98 91 6C E6 6C d8 02 181
0A28- E6 6d 56 77 68 68 05 73 190
0A30- 49 FF 85 73 28 0b 13 A5 1d3
```

SUM 5A 69 2b 45 29 7E 73 78 1d3

```
P13: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0A38- 73 10 21 42 15 45 52 1bE
0A40- 52 4F 52 08 20 42 15 54 1bE
0A48- 4F 4F 28 4d 41 AE 59 20 11C
0A50- 49 44 45 AE 54 49 46 49 14C
0A58- 52 45 53 80 A8 00 85 6b 17A
0A60- 91 6C E6 6b 18 F8 45 1CF
0A68- 4E A5 65 85 4C 98 02 E6 1A8
0A70- 6d A5 4C 9F 1b F0 CA 20 148
0A78- 0b 13 C9 36 16 28 0b 18E
0A80- 13 08 0b 05 6A 18 65 6b 192
0A88- 08 01 85 0b 28 0b 13 A9 1E8
0A90- 3A 28 0b 13 C9 38 96 183
0A98- 48 C9 32 d8 95 28 2E 0A 18F
0AA0- 08 F7 28 1E 0A A6 72 d8 1F7
0AA8- 02 06 6b C9 33 08 85 28 1E4
0AB0- 2E 0A 08 F7 46 65 28 C8 192
0AB8- 0A 24 68 10 0A 4C 3b 0A 159
0AC0- 28 C8 0A 24 80 10 F9 68 1FF
0AC8- 49 08 05 47 85 08 A5 10A7
0AD0- C9 34 98 13 C9 38 0F 168
```

SUM CF d1 47 29 b9 92 17 2d 1FF

```
P14: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0AD8- 69 0A 48 20 0b 13 28 C0 193
0AE0- 0A 68 46 08 4C 0b 13 C9 1eB
0AE8- 50 d8 19 20 F1 0A 4C 0b 156
0AF0- 13 28 1F 15 03 4C 03 1C 1d5
0AF8- 68 20 42 15 49 4E 56 41 105
0B00- 4C 49 44 80 C9 51 d8 14 1F7
0B08- A5 81 d8 18 A5 72 F0 E9 1FE
0B10- 48 20 0b 13 68 C9 01 F0 150
0B18- 83 28 0b 0F 20 1F 15 03 114
0B20- 4C 21 80 60 85 8E A9 bA 143
0B28- 20 Ed 14 A9 E8 20 Ed 14 1d3
0B30- C6 8E d8 F7 A9 9A 20 Ed 1eB
0B38- 14 4C 0C 0b C9 02 F8 A4 136
0B40- C9 C8 98 03 AC C1 11 C9 183
0B48- 07 98 03 4C 22 0C AA A5 163
0B50- 75 48 A5 6A 48 E0 85 d8 1C9
0B58- 30 20 23 13 68 AA 68 85 185
0B60- 76 48 8A 48 24 71 38 18 165
0B68- 24 76 18 08 A9 69 20 18 124
0B70- 18 2E 56 14 A9 01 d8 20 1CC
```

SUM d7 8A 4d 5F A9 E9 9C Ab 144

```
P15: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0B78- 18 65 69 bA E8 9A 46 1b0
0B80- 71 24 76 38 19 A9 83 d8 10F
0B88- 17 E8 06 d8 07 20 19 13 12C
0B90- 02 d8 0C E8 84 F8 03 15C
0B98- 4C 3b 0A 20 0b 13 A9 08 128
0BA0- 48 A5 68 C9 2C d8 09 26 153
0BA8- 0b 13 28 4E 4C d8 0b 15b
0BB0- A9 34 20 0b 13 A9 0F 20 17C
0BB8- 0b 13 28 0b 8F 68 A6 d8 102
0BC0- 65 69 68 95 76 8A d8 10C3
0BC8- A9 FC 24 76 18 02 A9 08 102
0BD0- 85 6A 6A A9 09 E8 08 F0 19F
0BD8- 83 4C 6A 18 4C 4C 97 10 184
0BE0- CA d8 19 46 65 20 1F 15 1b2
0BE8- 0A d8 68 A8 8A A9 10 24 125
0BF0- 76 18 d8 48 85 66 A9 09 1CA
0BF8- 48 4C 98 10 CA d8 18 28 118
0C00- 1F 15 07 AA 68 05 12 68 14C
0C08- 85 13 A2 08 20 A1 18 20 12b
0C10- 1F 15 03 8A 91 12 60 A9 16d
```

SUM d1 d8 Cb 43 9A 96 06 12 11F

```
P16: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0C18- 0A d8 b5 A5 00 A9 FF 85 17b
0C20- 08 68 C9 5A b0 F5 C9 52 1C3
0C28- 98 F1 E9 51 d8 20 0b 13 1F1
0C30- 6A 8A CA d8 41 86 65 28 1F8
0C38- 61 16 28 0C 0A 68 A9 28 18A
0C40- 0b 13 28 0b 8F 48 08 A5 1C4
0C48- 78 A5 7E A4 7F 18 02 141
0C50- 90 81 C8 38 Fd FE 01 85 112
0C58- 0C 98 F4 FF 01 d8 E5 164
0C60- 0C 18 0A CA 28 78 AA 162
0C68- A9 F8 4C E6 14 28 1F 15 133
0C70- 02 08 03 4C 48 16 CA F8 139
0C78- 83 4C 5A 8d E6 81 A6 48 127
0C80- E0 8A 98 2F 20 42 15 13 1C4
0C88- 49 4d 58 4C 45 28 56 41 12E
0C90- 52 28 45 58 50 45 43 34 136
0C98- 45 44 80 46 28 0b 0F 1F0
0CA0- A9 64 28 0b 13 A9 48 28 1dC
0CA8- Ed 14 28 61 16 46 45 28 163
0CB0- bA 68 A5 6A 48 A5 75 191
```

SUM 42 85 88 F9 8A F4 d2 FF 117

```
P17: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0CB8- 48 8A 28 E8 8b A5 68 C9 121
0CC0- 62 F0 4E A9 61 28 0b 13 168
0CC8- 28 98 0C 68 85 76 68 85 117
0CD0- 69 28 1F 15 82 bA E8 A9 18A
0CD8- FC 24 76 18 02 A9 08 85 14E
0CE8- 66 28 03 10 28 1F 15 83 158
0CE8- d4 88 01 24 76 38 8E 28 166
0CF0- 1F 15 02 0b 85 A9 CA 28 19E
0CF8- 6A 18 4C 07 d8 28 1F 15 12E
0D00- 86 0b 07 69 81 0A 20 1E2
0D08- 16 6A 9A C6 81 0C Ed 121
0D10- 14 20 0b 13 28 0b 4C d8 131
0D18- 85 76 88 05 69 A9 68 28 182
0D20- Ed 14 A9 FC 24 76 18 82 152
0D28- A9 88 85 6A A9 C9 20 65 193
0D30- 14 24 76 18 10 28 1F 15 11E
0D38- 09 0b 0A 08 b1 8A E9 08 1AF
0D40- 91 8A 4C 51 d8 20 1F 15 199
0D48- 83 08 06 A9 A9 CA 28 6A 1FE
0D50- 18 C6 81 48 48 16 CA d8 114
```

SUM 35 6A 15 A9 79 75 82 47 196

```
P18: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0D58- 2E 86 65 20 61 16 28 0b 156
0D68- 0F 8F 9A 28 0b 13 28 1F 119
0D68- 15 82 d8 83 28 55 16 05 11A
0D78- 65 85 A4 6A 48 20 bA 8A 10C
0D78- 68 85 6A 68 85 65 20 72 135
0D80- 16 28 48 16 4C 28 16 CA 1E8
0D88- 0b 4C 28 0b 8F A9 65 20 1Fb
0D90- 8b 13 28 1F 15 82 0b 83 1C7
0D98- 20 55 16 28 0b 86 A5 68 1FE
0DA0- C9 67 d8 0F 20 0b 13 28 114
0DA8- 55 16 28 72 16 28 28 1571
0DB0- 28 0b 0A 4C 28 16 A5 65 177
0DB8- 48 A5 6A 48 28 0b 6A 08 1E1
0DC0- AA 68 25 65 18 8A E4 6A 1F8
0DC8- F8 82 46 65 68 CA d8 4F 1E6
0DD0- 28 0b 0F A9 66 28 0b 13 187
0DD8- 28 08 08 0A 08 6A 67 161
0DE0- 28 95 0F A9 C9 2C 10 192
0DE8- 20 1F 15 02 F0 83 20 55 1bE
0DF0- 16 A5 65 48 A5 64 48 20 1d9
```

SUM E8 EF 8C F4 39 37 3F 85 196

```
P19: +0 +1 +2 +3 +4 +5 +6 +7 SUM
0DF8- bA 8A 28 55 16 28 72 16 1F3
0E00- 20 16 68 85 64 68 85 1FC3
0E08- 45 08 A5 A5 68 C9 67 d8 184
0E10- C9 8A 8A 28 0b 13 28 b6 15F
0E18- 0A 46 65 68 4C 1d 16 CA 166
0E20- F8 83 4C E5 8E A9 37 28 192
0E28- 0b 13 28 0F 20 0F 1F 15 14C
0E30- 02 95 16 A9 34 28 0b 13 141
0E38- 46 67 28 4E 8E A5 68 C9 1FF
0E40- 3C d8 86 20 0b 13 4C 38 184
0E48- 0E A9 3b 4C 0b 13 C9 4C 111
0E50- 08 25 20 1F 15 03 28 F7 163
0E58- 08 85 74 C9 22 F8 89 28 194
0E68- Ed 14 28 F4 13 4C 5b 8E 1d4
0E68- A9 80 46 65 28 Ed 14 20 195
0E70- F4 13 46 65 4C A5 13 C9 195
0E78- 6b 08 1d 20 0b 13 C9 37 14E
0E80- F8 88 20 1F 15 03 28 1A 189
0E88- 08 28 08 28 08 28 08 28 194
0E90- 20 1F 15 03 20 1F 01 60 177
```

SUM F1 2d 22 2d 6d 6d 33 6d 1E7



유틸리티

P:10 +0 +1 +2 +3 +4 +5 +6 +7 SUM
QE9B- C9 6A 40 8b 20 07 13 20 16B
QEAB- F1 53 03 20 29 81 60 C9 :2A
QEAB- 69 40 8b 20 07 13 20 1F 14D
QEAB- 15 83 20 84 16 68 C9 0A :A5
QEAB- 40 8F 20 19 13 20 1F 15 :7F
QECC- 85 AA 60 20 C4 00 4C E2 :A9
QECC- 8E C9 6D 40 0b 20 07 13 :59
QEED- 20 1F 15 83 20 A1 81 60 :F9
QEED- 20 0b 0F 20 1F 15 03 20 :31
QEED- C1 8E 46 65 60 C4 F0 83 :89
QEED- 4C 76 0F 49 37 20 0b 13 :6F
QEED- F8 15 20 42 15 43 4F AE :5C
QEED- 53 54 41 AE 54 20 45 50 :47
QEED- 50 45 43 54 45 44 00 A5 :5A
QEED- 6A 40 20 0b 13 A9 3C 20 :A5
QEED- 0b 13 48 AE 65 A4 48 20 :C3
QEED- 0F 10 13 42 00 0A 65 68 :4F
QEED- C9 30 F8 22 A9 3C 20 0b :A6
QEED- 13 20 0b 0F 60 A8 E8 40 :A7
QEED- 84 0E E9 20 42 15 54 4F :47
SUM b9 bb 06 87 CC 58 6E 62 :6D

P:11 +0 +1 +2 +3 +4 +5 +6 +7 SUM
PF3B- 4F 20 44 41 AE 59 20 50 :14
PF4B- 41 52 41 44 53 00 68 FC :CC
PF4B- 0F 0A 80 88 08 F0 89 B9 :83
PF5B- 71 0F 20 42 14 00 40 F7 :F0
PF5B- 20 1F 15 82 00 20 68 05 :6B
PF6B- 0C 68 AA A5 8C 20 E6 14 :E9
PF6B- A9 20 20 E6 14 46 45 4C :E9
PF7B- 0b 13 48 40 68 A9 20 1F :2F
PF7B- 15 84 0A 00 C0 C9 83 40 :A6
PF8B- 06 0A 10 0C 40 83 1C 06 :2E
PF8B- 20 0b 13 20 95 0F 20 56 :28
PF9B- 10 60 20 0b 13 20 03 0F :40
PF9B- A5 68 C9 90 90 F3 C9 9F :51
PFAB- 0B 5F 40 20 0b 13 20 05 :A9
PFAB- 8F 60 20 0b 13 20 2F 10 :89
PFAB- 4C 90 0F 20 C0 10 A5 68 :FF
PFAB- C9 00 90 05 40 10 80 A1 :90
PFAB- 48 20 0b 13 20 C0 10 68 :9D
PFAB- 48 4A 98 21 A9 C9 28 2F :8A
PFAB- 10 60 20 0E 14 20 1F 15 :E0
SUM 04 33 CB 15 54 F8 62 0E :92

P:12 +0 +1 +2 +3 +4 +5 +6 +7 SUM
QF0B- 07 0A A9 00 8F 02 A9 FF :AE
QF0B- 4C 66 0F A9 18 20 E4 1F :F3
QF0B- A9 E9 4C 2F 10 68 C9 8C :5A
QF0B- 0B 07 20 E3 0F A9 0b 0E :A4
QF0B- 49 C9 0A F0 13 A9 20 16 :1
QF0B- 2C 10 82 1F 15 07 50 0A :EB
QF0B- 10 00 30 0F 30 45 0F AA :A4
QF0B- 20 E3 0F 20 1F 15 07 50 :B4
QF0B- 8A 3C 00 10 82 1F 15 07 :F7
QF0B- 0F 20 1F 15 03 05 0C 68 :5F
QF0B- 20 08 13 20 48 18 24 :A7
QF0B- 13 30 32 48 20 21 10 68 :d4
QF0B- 18 69 FC 42 0C 4C E6 14 :71
QF0B- 48 C9 69 F0 8B 0E 40 :F2
QF0B- 0B A9 89 E9 51 86 00 20 :2A
QF0B- E2 14 A6 86 68 68 24 71 :11
QF0B- 10 F8 24 67 0E A9 4B :9C
QF0B- 20 E2 14 A9 A9 46 71 30 :62
QF0B- 66 67 48 A5 66 F0 08 C9 :E1
QF0B- 08 F8 24 C9 11 90 0b 66 :37
SUM db 3C 9E 72 6E 08 24 9F :C8

P:13 +0 +1 +2 +3 +4 +5 +6 +7 SUM
107B- 69 68 10 A5 66 66 71 AC :b7
108B- E6 14 A5 7A 18 65 69 AA :A9
108B- A5 7b 69 00 85 00 60 20 :A3

109B- 7A 10 A5 8d 4C Ed 14 20 :A9
109B- 9F 10 42 0A 4C 79 10 A6 :d6
10AB- 69 48 24 65 10 0E BA 30 :1A
10AB- E5 6A F0 12 C9 81 F0 10 :15
10AB- C9 FF F0 12 86 64 38 64 :52
10AB- A5 A9 40 20 E6 14 68 0A :98
10CB- E6 6A A9 C0 0A A9 08 :C0
10CB- C6 64 20 E2 14 68 60 20 :33
10AB- E2 10 A5 68 C9 86 F0 04 :44
10AB- C9 8E 08 F2 48 20 0b 13 :4F
10CB- 20 E6 10 68 18 69 10 20 :35
10CB- 29 10 4C 42 18 20 AE 11 :E6
10FB- A5 68 C9 08 F0 04 C9 09 :A4
10FB- d8 4A 20 0b 13 20 AE :48
110B- 11 24 71 30 1F 20 1F 15 :A9
110B- 02 AA 68 68 C9 89 F0 0A :C8
111B- 20 1F 15 83 20 64 80 4C :A7
SUM 4C F7 AA 2b 0b 64 8A 1C :E2

P:14 +0 +1 +2 +3 +4 +5 +6 +7 SUM
111B- F0 10 20 1F 15 83 20 05 :FC
112B- 80 4C F0 10 A5 66 F0 06 :C4
112B- C9 08 F0 0b A9 08 10 69 :FE
113B- 4C 20 65 10 4C 8b 11 A9 :48
113B- A6 20 E4 14 20 63 10 68 :C6
114B- C9 88 F0 CC 20 1F 15 03 :A4
114B- 20 8A 00 4C F0 10 A5 68 :93
115B- d0 14 48 20 56 10 A5 6A :C1
115B- 85 69 68 05 66 A5 71 A9 :A0
116B- FF 85 71 4C 8b 13 C9 80 :58
116B- F0 0A C9 81 d8 08 E9 81 :90
117B- 85 6A 40 80 F0 4C 10 :45
117B- F0 0A C9 19 d0 06 85 6A :9F
118B- A9 FC 00 CE C9 0A d0 8C :EC
118B- 24 75 10 83 A9 00 2C A9 :32
119B- FC 4C 52 11 C9 35 90 1E :57
119B- C9 38 0A 14 48 20 92 0F :44
119B- 85 6A 40 8A 4C 8b 13 :FF
11AB- 56 18 24 67 10 87 A9 :48
11AB- 20 E4 14 46 67 60 C9 83 :FA
SUM 97 34 A1 AE 2C 13 bc F3 :08

P:15 +0 +1 +2 +3 +4 +5 +6 +7 SUM
11AB- F0 0A C9 C0 80 74 20 87 :A8
11CB- 11 A5 6A 48 A6 6E F0 51 :51
11CB- 50 20 0b 13 C9 37 60 27 :34
11DB- A5 6b 48 E6 66 E6 6B :FA
11DB- 40 20 80 8F 46 67 66 6b :d6
11EB- 20 A1 10 20 1F 15 02 91 :89
11EB- 8A A5 68 C9 C0 F8 E6 68 :5A
11FB- 85 6A 49 3b 20 0b 13 30 :C4
11FB- 66 67 A9 A8 6B 20 E6 :24
120B- 14 46 65 68 C9 C0 8b 83 :63
120B- 4C F6 15 A0 FE E9 bF AA :47
121B- C0 C8 CA 00 Fb 20 15 :75
121B- b9 6b 17 5E b9 17 4C E6 :48
122B- 14 20 42 15 54 4F AE :94
122B- 4d 41 4E 59 20 41 52 47 :2F
123B- 53 08 C9 78 48 0b 20 07 :8E
123B- 13 20 1F 15 83 20 3b 08 :45
124B- 68 C9 71 d0 07 20 57 12 :FA
124B- E2 d4 81 68 C9 72 48 :d6
125B- 20 57 12 82 35 81 68 20 :11
SUM 85 94 5F 8F 98 0C 5d 8C :14

P:16 +0 +1 +2 +3 +4 +5 +6 +7 SUM
125B- 07 13 20 1F 15 03 85 17 :0D
126B- 20 4C 1F 15 C9 73 48 07 :b3
126B- 28 5F 12 82 6F 10 68 C9 :A4
127B- A8 98 31 C9 A7 8A 24 48 :FA
127B- 20 07 13 68 AA 4A 8A 0b :08
128B- 83 4C E0 14 C9 A8 09 :A8
128B- 98 11 20 1F 15 02 49 FF :3F

129B- 68 20 1F 15 05 18 49 FF :19
129B- 69 81 68 20 1F 15 04 AA :CC
12AB- 0A 8A 6A 68 C9 85 d8 47 :43
12AB- A5 6A 48 A5 75 48 28 23 :FC
12AB- 13 48 05 74 68 24 71 10 :85
12AB- 10 18 45 65 89 65 47 A2 :8A
12CB- 24 76 18 02 E2 86 66 46 :42
12CB- 68 46 65 24 76 38 0b A2 :82
12AB- 10 86 66 85 69 A9 A8 4C :87
12AB- 10 82 02 06 66 66 AA A9 :59
12EB- 69 20 48 10 20 E6 14 28 :13
12EB- 1F 15 83 20 0b 10 8A 68 C9 :C3
12FB- 06 40 4b 20 19 13 A2 00 :8F

SUM b7 96 bb 3E bc E7 EE F4 :C8

P:17 +0 +1 +2 +3 +4 +5 +6 +7 SUM
12FB- 20 A1 10 20 1F 15 05 87 :b1
130B- 12 68 05 13 b1 12 68 A9 :dE
130B- 36 40 20 0b 13 A9 37 20 :17
131B- 0b 13 20 0b 0F 68 4C 8b :97
131B- 13 A9 3C 20 0F 13 A9 36 :18
132B- 48 08 C9 E2 0b 13 A9 36 :d4
132B- 20 0b 13 28 95 0F 24 71 :17
133B- 10 87 85 66 F8 83 28 56 :b8
133B- 10 89 3A 40 0b 13 C9 05 :34
134B- 10 20 0b 13 45 A5 68 d0 :55
134B- 0A 6A 68 20 95 12 85 6A :CF
135B- 4C 72 11 28 66 11 28 56 :C2
135B- 10 48 91 12 C9 74 48 0E :1A
136B- 20 A7 11 20 0b 13 20 Eb :d1
136B- 0E A9 4C Ed 14 20 42 :10
137B- 10 89 3A 40 0b 13 C9 05 :45
137B- 20 46 41 43 54 4F 52 :00
138B- 0A 2A 2A 2A 85 8C 20 0F :51
138B- 45 8C 68 C5 68 F0 2C 20 :3A
139B- 42 15 55 4E 45 58 50 A5 :2C
SUM d0 bF A5 d8 10 d0 9E 2A :59

P:18 +0 +1 +2 +3 +4 +5 +6 +7 SUM
139B- 45 54 45 44 20 49 44 :12
13AB- 4E 54 00 C9 30 98 C0 :08
13AB- 3A 98 0C C9 41 90 8A C9 :3d
13BB- 5b 90 81 38 A2 FF 68 38 :56
13BB- E2 00 68 20 C1 13 45 68 :83
13CB- 08 A9 08 05 68 65 A5 :8A
13CB- 7A C9 21 90 89 C9 2E F0 :d4
13DB- 0A C9 3b 06 86 20 F4 13 :85
13DB- 4C C9 13 C9 25 40 89 20 :0F
13EB- F4 13 C9 20 8b F9 9E Ed :16
13EB- C9 27 00 8E 20 F4 13 85 :7A
13FB- 6A 28 F4 13 20 70 17 85 :d6
13FB- 74 48 C9 24 d8 21 28 F4 :C6
140B- 13 20 83 13 F0 80 0b F1 :82
140B- C9 47 d0 E9 E9 86 9F 80 :85
141B- 85 0C A5 6A 8A 0A 0A :C8
141B- 18 65 8C 85 6A 90 4F :87
142B- A3 13 48 49 E9 38 0A A5 :87
142B- 68 28 28 F4 13 20 85 6A :86
143B- 18 98 65 6A 65 6A 20 F4 :82
SUM 25 Ab bA 45 6E F9 97 78 :d5

P:19 +0 +1 +2 +3 +4 +5 +6 +7 SUM
143B- 13 28 A3 13 F0 E6 68 48 :67
144B- 89 80 80 81 30 5E 14 68 :24
144B- 20 A3 13 bA 24 A5 74 28 :E3
145B- A3 13 90 82 08 12 44 6E :3C
145B- 20 5E 14 4C 14 91 6C :3C
146B- C0 84 4E 08 8F 4C 3b 8A :AA
146B- A5 73 18 85 49 FF 85 73 :6d
147B- 68 A5 77 81 80 A5 15 85 :FF
147B- 12 A5 6d 85 13 A5 8A :82
148B- 6E 88 28 97 14 b1 6C 81 :9F
148B- 12 F0 30 A1 12 F0 17 38 :1C



SUM 18 34 CA 4C 7b 29 9b 82 :23

리스트 4: TL2의 덤프 리스트

P:1 +0 +1 +2 +3 +4 +5 +6 +7 SUM
7E00- 20 57 FF bA bd 00 01 AA +9
7E09- A9 00 E0 86 0C 95 06 E0 +9
7E10- 86 0A 0A b1 0b F0 1F +Ad
7E19- 85 0d C0 b1 0b 18 A5 0A +9d
7E20- 85 0E C0 b1 0b 18 65 0A +9d
7E29- 01 0d C0 d0 E7 05 0E C0 +8d
7E30- b1 0b 18 65 0A 01 0d C0 +99
7E39- b1 0b 05 0d C0 b1 0b d0 +A2
7E40- EC A9 34 85 0b A9 02 18 +1C
7E49- 65 0A 85 0C A9 02 18 65 +20
7E50- 0A 0A 01 91 8b C0 C0 +97
7E59- 41 00 F0 68 00 93 02 02 +00
7E60- b6 02 00 b9 02 00 b7 02 +31
7E69- 02 F0 02 00 00 85 09 00 +0A
7E70- 4d 07 00 50 09 00 02 09 +3A
7E79- 03 EC 09 00 22 0b 00 50 +7F
7E80- 0E 00 80 0E 01 96 0E 01 +4A
7E89- A5 0E 01 b4 0E 01 C5 0E +4A
7E90- 00 06 0E 01 E1 0E 00 16 +EA
7E99- 11 00 20 11 00 0A 11 00 +9d
SUM A4 95 F0 E0 25 7C F0 EC +A6

P:2 +0 +1 +2 +3 +4 +5 +6 +7 SUM
7E40- 3F 12 00 24 10 00 45 +F3
7E49- 01 4F 15 01 52 15 01 55 +23
7E50- 15 01 5A 15 01 63 15 01 +FF
7E59- 09 15 00 0C 15 00 92 15 +EA
7E60- 01 95 15 00 A7 16 00 +12
7E69- 16 00 C2 16 00 CF 16 00 +d3
7E70- 0C 17 01 2F 17 01 4C 17 +C0
7E79- 01 69 17 01 6C 17 01 7d +93
7E80- 17 01 69 17 01 69 17 01 +81
7E89- bb 17 02 bd 17 02 bf 17 +80
7E90- 02 00 00 00 00 1C A4 +92
7E99- 5b b9 09 0d 30 0C 18 65 +63
7F00- 42 00 00 53 00 00 60 00 +F5
7F09- 00 b2 00 00 d2 00 00 E9 +6d
7F10- 00 01 EF 00 01 FF 00 01 +F1
7F19- 15 01 01 15 01 01 24 01 +5C
7F20- 01 50 01 01 40 01 01 4A +bF
7F29- 01 00 01 01 01 5E 01 00 +AF
7F30- 64 01 01 74 01 01 9A 01 +77
7F39- 01 9F 01 01 A9 01 01 C1 +0d
SUM EF E1 29 5E 64 02 6d +C8

P:3 +0 +1 +2 +3 +4 +5 +6 +7 SUM
7F40- 01 82 C6 01 02 C4 01 0b +7b
7F49- dE 01 82 E3 01 02 E0 01 +0b
7F50- 01 F7 81 00 14 02 01 17 +27
7F59- 02 00 76 02 00 93 02 +11
7F60- b6 02 00 b9 02 00 0C 02 +31
7F69- 02 F0 02 00 00 85 09 00 +0A
7F70- 4d 07 00 50 09 00 02 09 +3A
7F79- 03 EC 09 00 22 0b 00 50 +7F
7F80- 0E 00 80 0E 01 96 0E 01 +4A
7F89- A5 0E 01 b4 0E 01 C5 0E +4A
7F90- 00 06 0E 01 E1 0E 00 16 +EA
7F99- 11 00 20 11 00 0A 11 00 +9d
7FA0- 3F 12 00 24 10 00 45 +F3
7FA9- 01 4F 15 01 52 15 01 55 +23
7FB0- 15 01 5A 15 01 63 15 01 +FF

7FB9- 09 15 00 0C 15 00 92 15 +EA
7FC0- 01 95 15 00 A7 16 00 +12
7FC9- 16 00 C2 16 00 CF 16 00 +d3
7FD0- 0C 17 01 2F 17 01 4C +1CE
7FD9- 01 69 17 01 6C 17 01 7d +83

SUM b0 59 5F d8 dA d0 6E 61 +C1L
P:4 +0 +1 +2 +3 +4 +5 +6 +7 SUM
7FE0- 17 01 60 17 01 b9 17 01 +B1
7FE9- bb 17 02 bd 17 02 bf 17 +80
7FF0- 02 A4 12 01 55 12 01 FA +C1
7FF9- 14 00 6d 12 01 00 00 00 +94
8000- A9 00 A2 00 05 0A 06 0b +6b
8009- d0 4C 03 1C A5 0A 91 0A +8d
8010- C0 A5 0b 91 0A 30 90 +54
8019- 0A 85 0A 90 02 E6 0b 60 +7C
8020- d0 40 30 A5 0A E9 02 05 +77
8029- 0A 00 02 C6 0b A0 01 b1 +dF
8030- 0A AA 00 b1 0A 05 0A 06 +0C
8039- 0b 00 60 40 40 05 0F A2 +7d
8040- 20 64 00 10 69 01 90 01 +17
8049- E0 95 0F 0A 40 15 02 +A5
8050- 7d 20 A4 00 05 0C 60 10 +92
8059- 65 0C AA 05 10 60 20 64 +9C
8060- 00 E0 8A 00 05 0C AA F0 +5d
8069- 19 CA 06 0d A2 00 A9 00 +C9
8070- 66 0C 90 02 05 0d 6A 66 +46
8079- 0C CA 00 F6 05 18 AA +50
8080- 00 00 00 00 00 00 00 00 +00
SUM 27 7F EA 94 bF 6F AF 3F +40

P:5 +0 +1 +2 +3 +4 +5 +6 +7 SUM
8080- 0C 60 05 18 60 05 0E 0A +86
8089- 00 05 06 0E AA F0 1F A2 +C4
8090- 00 0A 0C E0 8A 90 FC 05 +95
8099- 0d A5 0E 66 0d C5 0d 90 +95
80A0- 02 E5 0d 26 0C CA 08 F3 +b3
80A9- 05 19 AA A5 0C 60 05 16 +F4
80B0- 20 F7 00 0d 07 45 52 52 +94
80B9- 20 44 49 56 30 00 4C 03 +82
80C0- 1C AA A9 00 05 11 0A A0 +2F
80C9- 00 05 0E A5 19 40 A2 0A +A5
80D0- 20 07 00 05 0E A0 00 C0 +5A
80d0- A5 0E 0d F2 A5 11 F0 0A +25
80E0- 84 11 38 E5 11 90 03 20 +76
80E9- 29 01 60 07 30 20 b4 01 +A0
80F0- 08 00 F7 60 05 19 60 08 +1d
80F9- 05 12 60 05 13 20 0d 01 +A5
8100- 30 90 45 12 05 12 90 02 +78
8109- E6 13 4C 12 00 A0 01 b1 +C9
8110- 12 F0 06 20 b4 01 C0 d0 +F5
8119- F6 60 A9 0d 4C b4 01 AA +37
SUM 71 FC 2b EA 1F Fd Bb 42 +Fb

P:6 +0 +1 +2 +3 +4 +5 +6 +7 SUM
8120- F0 F7 20 1A 81 CA 00 FA +36
8129- 60 AA F0 Ed A9 20 20 b4 +84
8130- 01 CA 00 FA 60 A5 19 40 +7C
8139- A5 18 40 A9 00 40 20 d4 +EA
8140- 01 C9 00 40 09 60 A2 0A +3F
8149- 20 05 00 4C 5d 01 3E E9 +50
8150- 30 90 12 C9 BA 00 0E 05 +E0

8159- 11 60 A2 0A 20 64 00 18 +41
8160- 65 11 4C 3d 01 60 AA 60 +FA
8169- 05 18 40 05 19 0A 00 49 +34
8170- 00 40 20 d4 01 C9 00 40 +5E
8179- 07 60 AA 4A 4A 4A 18 F1 +98
8180- 30 E9 30 90 16 C9 0A 90 +5A
8189- 06 E9 07 C9 10 b0 0C 05 +10
8190- 0C 60 8A 0A 0A 0A 05 0C +Ad
8199- 4C 71 01 60 60 20 A1 01 +48
81A0- 0A 40 4A 4A 4A 4A 20 AC +C6
81A9- 01 60 29 0F 09 30 C9 3A +5d
81b0- 90 02 49 06 05 11 40 6A +69
81b9- 40 90 40 A5 16 0A AA b0 +54

SUM C2 97 68 40 d0 11 4A Fb +3C
P:7 +0 +1 +2 +3 +4 +5 +6 +7 SUM
81C0- 54 02 05 14 b0 55 02 05 +00
81C9- 15 A5 11 20 F2 01 60 A0 +AE
81D0- 60 AA 60 60 8A 40 90 40 +8C
81d0- A5 17 0A AA b0 34 02 05 +60
81E0- 14 0d 35 02 05 15 20 F2 +34
81E9- 01 95 11 60 40 60 6A A5 +dE
81F0- 11 60 40 14 00 20 00 00 +94
81F9- A0 00 b1 0A 05 15 C0 b1 +6E
8200- 0A 05 14 C0 b1 0A 05 8C +b7
8209- C0 b1 0A A0 C0 b1 0A A0 +50
8210- A5 0C 20 F2 01 60 20 00 +30
8219- 09 00 4C E0 F4 20 35 F4 +11
8220- C9 95 00 02 01 20 C9 E0 +b2
8229- 90 02 29 6F 40 20 6d F4 +EC
8230- 60 29 7F 60 1d 02 1d 02 +AE
8239- 1d 02 1d 02 1d 02 1d 02 +7C
8240- 1d 02 1d 02 1d 02 1d 02 +7C
8249- 1d 02 1d 02 1d 02 1d 02 +7C
8250- 1d 02 1d 02 1d 02 1d 02 +7C
8259- 18 02 18 02 18 02 18 02 +60
SUM 09 96 F9 62 3C 7F E0 4C +F1

P:8 +0 +1 +2 +3 +4 +5 +6 +7 SUM
8260- 18 02 18 02 18 02 18 02 +60
8269- 18 02 18 02 18 02 18 02 +60
8270- 18 02 18 02 20 0C 00 A0 +00
8279- 00 0A 0E 0A 18 b1 0A 05 +AE
8280- 0d C0 b1 0A 05 0C C0 b1 +9A
8289- 0A 05 13 C0 b1 0A 05 12 +bC
8290- 60 20 74 02 1d 40 0A 04 +FC
8299- 06 0E 26 18 0A 0C 26 0d +97
82A0- 90 0A 10 05 12 05 0E 05 +A4
82A9- 0E A5 13 05 18 05 18 CA +AA
82B0- 0d E6 A5 0E 4C 21 00 4C +A2
82B9- AE 00 20 74 02 05 13 F0 +4C
82C0- F6 A2 11 CA 06 0C 26 0d +b0
82C9- 9F 20 6E 26 18 05 AE +AE
82D0- C3 12 A5 18 E5 13 90 07 +25
82d0- 05 10 A5 0E 05 12 05 0E +6A
82E0- 30 26 0C 26 0d CA 00 E2 +19
82E9- A5 18 05 A5 A5 0E 05 19 +Ad
82F0- A5 0d 05 18 05 0C 4C 21 +Ad
82F9- 00 A5 1A 00 1F 15 07 05 +5F
SUM b3 52 55 b0 AA 45 3E 5b +9A