## A TINY SUGGESTION: A *LIST* OPTION FOR MULTIPLE–STATEMENT–PER–LINE BASICS

12/29/76

Li-Chen Wang's excellent implementation of Tiny Basic allows multiple commands per line. This capability is useful but makes the logic of a Basic program hard to follow in a listing, especially for people unfamiliar with the program and/or the language. So it might be kind of nice to have an optional variation of the LIST command that would list a program one command per line. This shouldn't add much to the interpreter, since most of the code required is there already. Simply replacing all semicolons with carriage returns in the LIST output would be one way of doing it. (Yes, I know. Some strings would look funny.) Maybe such a Modified-LIST command could be called MIST, which could be abbreviated M.

Jim Day

## A RESIDENT PL/M COMPILER FOR THE 6800 – SOON(?)

Dear Jim: Dec. 30, 1976

In Volume 1, Number 10 of *DDJ* Larry Walker put a high priority on a procedure-oriented language that can be compiled. I agree, and thought I would let you people know that for the last several months I have been working on a 6800-resident PL/M compiler.

I chose PL/M since it is ideally suited to systems programming; in addition it appears easily extendable for other applications (more about that later). One of the best things about PL/M is that it can be compiled in only one pass; this means you can have the source coming in off one cassette drive, and the object going out to another with no intermediate files.

So far I have completed the overall design of the compiler, plus coding of the lexical scanner, parser, and symbol table routines. I am currently writing the code-generation routines. For any compiler freaks out there, I am using operator precedence for parsing arithmetic expressions, and recursive descent for everything else.

I am writing the compiler in a subset of PL/M, and then hand translating to assembly language; once the first version is up and running it will be able to compile itself. The subset includes only those features of the language necessary to write the compiler. Notable exceptions are the lack of parameters in procedures (using global variables instead); simplified expressions; no based variables; and no GOTO statements. Later I will add these features incrementally, until the full language is implemented.

Since I have a Sphere 330 system, the first released version will be for that machine. Although the 330 has a 20K RAM, I am hoping the compiler will fit in a 16K system. The code for the initial (subset) version looks like it will be around 6K. After the Sphere version is up I plan on releasing a MIKBUG version. Hope to have something available by this summer.

Then come extensions, such as:

1) carriage control and a title capability
2) qualified variables and structure declarations down to the bit level
3) "real" macros, implemented as %procedures (as compared to the DECLARE LITERALLY)
4) multiple-dimensioned arrays
5) floating point arithmetic (probably TSC format, i.e. 9 significant digits with exponent range -99 to +99)
6) varying character strings, and string functions

It is probably not feasible to implement all of these in a single version of the compiler, and still keep the size reasonable. I have already been involved in adding the first three extensions to Intel's PL/M compiler via a SPITBOL preprocessor; the resulting language has been used for the past year in writing control software for an 8080.

Note that I am not proposing formatted I/O, even though PL/M supports only single character input/output. I feel that formatted I/O can best be handled via a library of suitable procedures, called at run-time.

I welcome suggestions from other PL/M users regarding possible extensions.

Sincerely yours,

Thomas W. Crosley
14-3 King Arthur Ct.
Northlake, IL 60164

## Tiny BASIC on a Honeywell 112 in Europe

Dear Sir, Dec. 15, 1976

While visiting America recently I had the opportunity to buy several copies of *Dr. Dobb's Journal of Computer Calisthenics and Orthodontia* and enjoyed very much the general style of the magazine and the interesting software listings.

I don't own an 8800 or 6800 but a special 12 bit machine with a similar instruction set to PDP-8, however it usually doesn't prove too difficult to convert most of the programs. I was especially interested in Tiny BASIC for my machine with one or two additions for ASC11 character string handling.

My system comprises a 12 bit Honeywell 112 mini with 8k of core memory, a creed 5 unit keyboard/printer/paper tape punch plus a separate 5 unit creed paper tape reader, cassette tape working at 3,300BPS which allows loading of 4k in under 15 seconds a Digital group video display (still under construction) and a South West Tech. keyboard (for ASC11 input).

Yours sincerely,

Ian D. Spencer
Am-Museum 3,
5353 Mechernich
West Germany

## 6502 RESIDENT ASSEMBLER PROGRAM AND TINY BASIC INTERPRETIVE PROGRAM NOW AVAILABLE IN ROM FOR $200

Santa Clara, California, April 1, 1977 . . . Microcomputer Associates Inc. today announced shipment of their Resident Assembler Program (RAP) and Tiny BASIC interpretive program on ROM chips. Two 2K x 8 ROMs comprise the software ROM package, housing the 1.75K Resident Assembler and the 2.2K Tiny BASIC program.

Formerly contained in seven 1702A PROMs, RAP is the only single pass Resident 6502 assembler available today. Statements are entered either from paper tape or directly from a terminal keyboard. RAP generates a listing and places object code into RAM for immediate execution. A minimum of 4K x 8 RAM memory is needed with the users' 6502 microcomputer. RAP allows a 5602 microcomputer to function economically as a microcomputer development system. Following assembly the programs can be debugged using the debugging facilities of DEMON, Microcomputer Associates' DE-bug MONitor program housed in the 1K ROM section of a 6530 ROM/RAM-I/O-Interval timer circuit. A text editor is included.

For those microcomputer users who prefer to use a somewhat higher level language Tiny BASIC permits immediate entry and execution of Tiny BASIC language programs. Statements include: LET, IF . . . , THEN, INPUT, PRINT, GOTO, GO SUB, RETURN, END, REM, CLEAR, LIST, RUN, and functions RND (Random Number generator) and USR (User SubRoutine) that allows branching, with arguments to assembly language subroutines. ROM software has been designed so that most I/O devices can be used.

The ROMs are totally pin-compatible with 2708-type PROMs.

The RAP/Tiny BASIC ROM package (SW101) is priced at $200 and includes full documentation with deliveries from stock to 30 days ARO.

RAP is also available on a set of seven 1702A PROMs (SW200) for $295. Tiny BASIC is available either in paper tape format (SW300) for $25 or on a set of nine 1702A PROMs (sw201) for $275. All software is fully documented with deliveries from stock to 30 days ARO.

For further information contact Darrell Crow, Microcomputer Associates, 2589 Scott Blvd., Santa Clara, CA 95050, (408)247-8940.