ALPHA DISC LTD
UNIT 2, CRABTREE ROAD, THORPE INDUSTRIAL ESTATE, EGHAM, SURREY. TW20 8RN
TELEPHONE: (0784) 35357/8/9 TELEX: 918886 ALPHAD G

Dear User

Thank you for your enquiry about Pineapple's Basic Compiler program. We have attempted to give a brief description of its facilities here, but if you have any further queries please 'phone the above number for more information.

The BASIC compiler translates a source file of statements, very closely resembling BBC BASIC, into a machine code file which the BBC micro computer is able to independently execute (*RUN) from a BASIC program at a later time.

Although the BBC BASIC interpreter is used at compile time, compiled programs will run on a machine without a BASIC ROM fitted.

BASIC was originally designed as an interactive language intended to be easily interpreted. It is therefore not very surprising that it is difficult to compile BASIC efficiently, compared with languages designed specially to be compiled. In spite of this, there are several sound reasons for seriously considering a BASIC compiler.

One advantage of a BASIC compiler is the fact that the language is so well understood by a large number of people. BBC micro users do not have to learn a totally new language in order to compile a progam.

Secondly, the means of entering and editing of program text is already available on the standard machine.

Finally, with trivial program changes (if any), much of the debugging can normally be done by interpreting before the final compilation, saving temper and time.

The reasons for compiling in the first place are usually speed and the ability to run independently of a language interpreter.

An interpreted program usually runs much slower because the interpreter must search through tables to find the location of procedure, function, variable names and line numbers as required during program execution.

Where a program is compiled all such scanning is done once at compilation time, relevant addresses being inserted into the target code. This gives the compiled program the speed advantage so important in time-critical applications.

Not counting the 16K BASIC interpreter ROM, the size of a compiled program is generally greater than the source program. This is not such a problem and does not prevent the compilation of the largest of programs provided a disc filing system is used. Since variable values are preserved when overlaying, a very large program may easily be segmented and sections 'swapped' from disc during execution.

In order to run at a reasonable speed most interpreters need a very large repertoire of built-in procedures and functions because user defined ones, being wholly interpreted, run more slowly.

The compiler attaches a package of frequently used mathematical and input/output routines to each compiled module to ensure compactness and autonomy.

To reduce the overhead this imposes, particularly with short programs, the usual strategy of making infrequently used routines, user defined, has been adopted.

To this end, the compiler has been designed to facilitate the incorporation of user defined library routines, which are added only if needed by a particular program.

Since these are to be compiled there will be no significant speed penalty. This effective extendability of the compiler is a very attractive feature.

The length of the compiler is critical since it is resident in memory while compiling, however, even with the considerable number of built in functions the compiler provides, the production of contiguous sections of machine code, up to 11K bytes long has been made possible.

It is worth observing that certain features of BBC BASIC such as floating point arithmetic and trigonometric functions for example, if incorporated, would not necessarily result in the final program running much faster when compiled.

The program will only work with a BBC micro computer fitted with a disc drive and either BASIC1 or BASIC2.

PTO

BASIC COMPILER FEATURES

KEYWORDS

The compiler supports all the following BBC Basic operators:-

! ? * DIV MOD - + = <> < > AND OR EOR

The following Basic commands are also supported:-

$ [ ] & \ ' : ~ . ; "

*FX

ABS ADVAL AND ASC CALL CHR$ CLG CLS COLOUR DEF DIM DIV DRAW ELSE END
ENDPROC ENVELOPE EOR FALSE FN FOR GCOL GET GOSUB GOTO IF INKEY INPUT LEN
LET LOCAL MOD MODE MOVE NEXT NOT ON OR PLOT POINT POS PRINT PROC REM
REPEAT RETURN RND RUN SGN SPC SOUND STEP STOP STR$ TAB THEN TIME TO TRUE
UNTIL USR VAL VDU VPOS
Any O.S. command (any statement starting with * eg *LOAD, *RUN, etc).

VARIABLES

Only integer variables A% toZ% are allowed but this is not so restrictive since parameters and local variables may be used in functions and procedures.

Two byte storage of variables is used allowing numbers between -32767 and +32767 to be used.

STRINGS

Strings are difficult to implement in the compiler environment but use has been made of the BBC's facility to use the 'string indirection operator,' and this method of storing strings has been fully implemented.

```
eg.  $A%="Pineapple"
     PRINTCHR$(F%)
     $G%=$(&600)+"Pineapple"+CHR$(E%)+STR$(H%)
     PRINT"Basic Compiler"
```

are all legal statements.

OTHER FEATURES

1. Parameter passing for functions and procedures is fully implemented.

2. Nesting of all loops and procedures is allowed.

3. Compiled programs up to 11K long may be produced directly in one operation. Longer programs can be produced by the use of overlay techniques.

4. Multiple file techniques allow the use of a disc library of predefined functions and procedures to be incorporated in any program, and discs will soon be available with library routines for such things as file handling etc.

5. Assembly language statements may be included in the Basic program to be compiled.