APPENDIX 3
FORTH-79 STANDARD

The purpose of FORTH-79 Standard is to allow transportability of standard FORTH programs in source form among standard FORTH systems. A program written according to the Standard will run equivalently on any FORTH system that adheres to the Standard.

The current Standard was developed by the FORTH Standards Team. (The Standards Team is not affiliated with FORTH, Inc., but the company does have three voting members on the team.) This Standard is a descendant of FORTH-78 (proposed by the FORTH International Standards Team) and before that of FORTH-77 (the work of an informal group of European and American FORTH users). Efforts at standardization go back as far as 1973, at Kitt Peak Observatory in Arizona.

Having voted to accept the FORTH-79 Standard, FORTH, Inc. revised its product line to adopt most of the Standard's features and naming conventions. Of course the Standard attempts to cover only a minimal system. Therefore it doesn't address many powerful words and features included in FORTH, Inc.'s polyFORTH, which represents the state-of-the-art in FORTH implementations. In this book we've included many words which we feel are likely to be adopted by future Standards.

A small number of issues raised by the FORTH-79 Standard remain controversial. In a few cases, the functions of words as described in this book don't follow the FORTH-79 Standard, but rather the FORTH, Inc. product line. Most of these discrepancies have been marked with footnotes; however, a few are more general in nature and deserve special discussion.

The most noticeable difference is in the length of the name field for each dictionary entry. The Standard specifies that dictionary entries include up to 31 characters of the name to avoid "collisions." FORTH, Inc. implementations use a count and three characters not only to save memory, but also to support dictionary search routines that are significantly faster than any 31 character implementation seen to date. FORTH, Inc. is presently researching algorithms which may offer users greater flexibility in naming, without unacceptable sacrifice in performance.

The FORTH-79 Standard includes a few words which change their behavior depending on a variable called STATE, which indicates whether the user is in "compile mode." One is $\boxed{."}$. In FORTH, Inc. implementations, $\boxed{."}$ is a compiling word, and therefore it may only be used inside a colon definition. In FORTH-79 languages, it has two functions: if the system is in execution mode, it will type the string which follows it at the terminal from which it was just entered.

A more significant controversy related to STATE is the behavior of the word $\boxed{'}$ (tick). In FORTH, Inc. languages, tick always reads the next word in the input stream when tick is executed. The Standard tick however, has two behaviors: when the system is in execution mode, it behaves in the normal way, but in compile mode it behaves like $\boxed{[']}$ (bracket-tick-bracket); that is, it compiles the address of the next word in the definition as a literal. To define a word which must "tick" the next word in the input stream when the word is executed, you must use the phrase

```
[COMPILE] '
```

if you're using the Standard tick .

There's one other difference worth mentioning here. The FORTH-79 Standard does not make the assumption that the $\boxed{DO}$ loop index and limit will be kept on the return stack. Presumably a system may have a third stack. For this reason, the Standard includes the word R@ to copy the top value from the return stack onto the parameter stack. In all systems that we know of, however, R@ would be identical to the FORTH $\boxed{I}$.

For more information or for copies of the FORTH-79 Standard, write to the FORTH Interest Group (FIG), P.O. Box 1105, San Carlos, CA 94070.