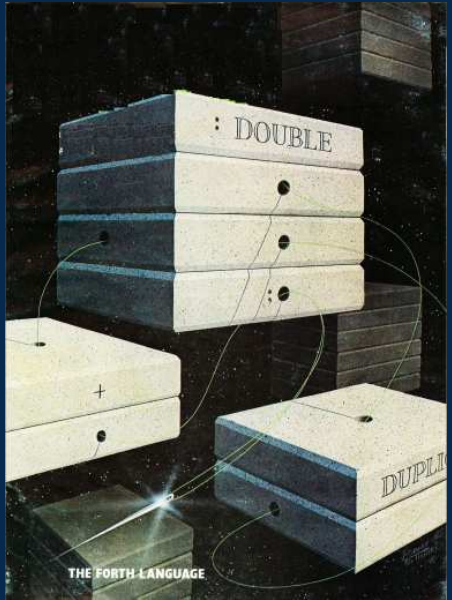


Use the Forth

Brian L. Stuart



Charles Moore: Creator of Forth



I like the observation that Forth is an amplifier: a good programmer can write a great program; a bad programmer a terrible one. I feel no need to cater to bad programmers.

(Charles H. Moore)

izquotes.com

Forth Background

- Developed on IBM 1130 at NRAO
- Major Applications:
 - Optical and radio telescope control
 - Satellite and space exploration systems
 - Large LASER array control
 - OpenBoot: Sun, IBM, Apple, ARM, OLCP XO-1
 - FreeBSD boot loader
 - Canon Cat
 - Jupiter ACE
 - Available for almost every small computer

Forth Background

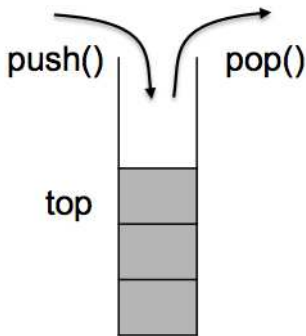
- “Standards:”
 - figForth
 - Forth-79
 - Forth-83
 - ANS Forth (ANSI '94)
- Forth processors
 - Harris RTX-2010
 - F21
 - PSC 1000
 - Novix NC4000
 - SEAforth24 and 40
 - GA 144

Forth Syntax

NONE!

- Words separated by spaces

Stack



stack



- Defined by `push()` and `pop()` operations
- Last in-First out (LIFO) structure

Forth Stack Use

- Numbers get pushed
- Words pop arguments and push results
- Examples
 - 3 .
 - 3 5 .
 - 3 5 + .
 - 3 5 XOR .

Basic Stack Operations

- . — Pop and print the top of the stack
- + - * / MOD ABS — Usual arithmetic operations
- MINUS — Unary negation
- AND OR XOR — Bitwise Boolean operations

Stack Manipulations

- DUP — Duplicate top of stack
- DROP — Pop and discard top of stack
- SWAP — Pop top two numbers and push in reverse
- ROT — Remove third element and push it on top
- OVER — Copy second element and push
- ! — Store second value at address on top
- @ — Fetch: pop address and push contents of the address

Number Bases

- DECIMAL — Switch to base 10
- HEX — Switch to base 16
- BASE — Variable holding current base
- Example:
 - 5 BASE ! — Set to base 5

Defining New Words

- `: ... ;` — Define a new executable word
- `VARIABLE` — Create a variable and word to address it
- `CONSTANT` — Create a named constant
- `VLIST` — Print a list of defined words
- Examples:
 - `: SQ DUP * ;` — Define a function to compute the square of the top of the stack
 - `VARIABLE TEMP` — Create a variable called `TEMP`
 - `5 CONSTANT FIVE` — Give the number 5 the name `FIVE`
 - `FIVE TEMP !` — Store 5 into the variable `TEMP`
 - `TEMP @ BASE !` — Use the value of `TEMP` to set the number base

Control Flow

- IF ... THEN — Simple condition
- IF ... ELSE ... THEN — Two-way branch
- DO ... LOOP — Count by 1 loop
- DO ... +LOOP — Arbitrary increment loop
- I — Push loop counter onto stack
- BEGIN ... UNTIL — Post-test loop
- BEGIN ... WHILE ... REPEAT — Pre-test (mid-test) loop

Examples

- : IFACT 1+ 1 SWAP OVER DO I * LOOP ;
- : GCD BEGIN SWAP OVER MOD DUP 0= UNTIL DROP ;
- : RFACT [SMUDGE] DUP 0= IF DROP 1 ELSE
DUP 1- RFACT * THEN [SMUDGE] ;

A Little Bigger Example

- Find all 3-digit numbers that are equal to the sum of the cubes of the digits
 - : CUBE DUP DUP * * ;
 - : DIGITS DUP DUP 100 / ROT 10 MOD ROT
10 / 10 MOD ROT ;
 - : 3CUBES CUBE ROT CUBE ROT CUBE ROT ;
 - : CUBIES 1000 100 DO I DIGITS 3CUBES + +
I = IF I . CR THEN LOOP ;