INF 212
FORTH

History of Forth

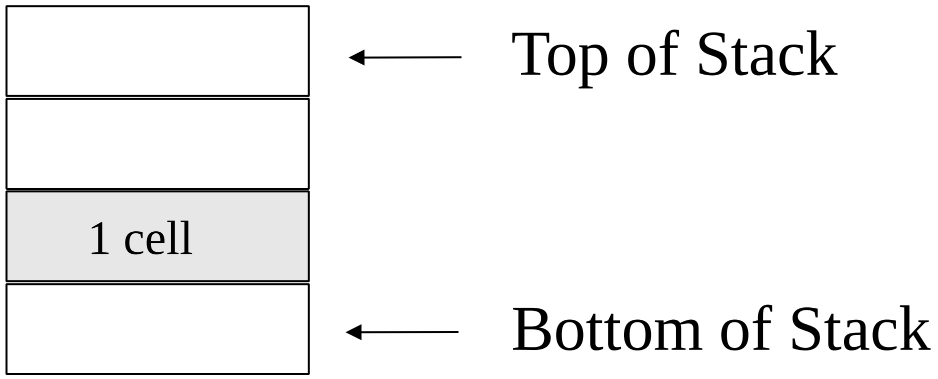
- Forth was developed by Chuck Moore in the 1960s (see Forth - The Early Years by C. Moore and The Evolution of Forth by E. Rather, et al).
- Original use for Forth was to perform instrument control, data acquisition, and least-squares curve-fitting at NRAO and Kitt Peak.
- Became a formal programming language in 1977 with Forth-77 standard. Subsequent standards were Forth-79 and Forth-83 by the Forth Standards Team.
- First commercial Forth system for IBM-PC introduced in 1982 by Laboratory Microsystems, Inc.
- Became an ANSI standard language in 1994, resulting in ANS-Forth.

Overview of Forth

- Forth is interactive
  - Perform computations directly at the Forth prompt.
  - Define and examine variables and constants
  - Define and execute new Forth *words* (individual subroutines).
  - Execute operating system commands.

Overview of Forth

- Forth syntax is derived from use of a data stack.
  - The basic method of passing arguments to, and obtaining results from, Forth words is through the data stack.



Overview of Forth

- Forth maintains a list of words, a ***dictionary.***

```
words
WORD        WORDS       FIND        '           [']
[           ]           CREATE      DOES>       >BODY
FORGET      COLD        ALLOT       ?ALLOT      LITERAL
EVALUATE    IMMEDIATE   CONSTANT    FCONSTANT   VARIABLE
FVARIABLE   CELLS       CELL+       CHAR+       DFLOATS
DFLOAT+     SFLOATS     SFLOAT+     ?           @
!           2@          2!          A@          C@
C!          W@          W!          F@          F!
DF@         DF!         SF@         SF!         SP@
RP@         >R          R>          R@          2>R
2R>         2R@         ?DUP        DUP         DROP
SWAP        OVER        ROT         -ROT        NIP
TUCK        PICK        ROLL        2DUP        2DROP
2SWAP       2OVER       2ROT        DEPTH       BASE
BINARY      DECIMAL     HEX         1+          1-
2+          2-          2*          2/          DO
?DO         LOOP        +LOOP       LEAVE       UNLOOP
I           J           BEGIN       WHILE       REPEAT
UNTIL       AGAIN       IF          ELSE        THEN
CASE        ENDCASE     OF          ENDOF       RECURSE
BYE         EXIT        QUIT        ABORT       ABORT"
                                                                  …
```

Applications of Forth

- Embedded Systems:
  - smart cards, robotics, Fed-Ex package trackers, embedded web servers, space applications
- Software Tools Development
  - writing cross-assemblers and disassemblers
  - writing parsers and programming languages
  - scripting and software testing
- Application Development
  - editors, word processors, games, circuit modeling, VLSI design, ...
- Laboratory Automation
  - Hardware Interfacing
  - Data acquisiton, data logging
  - Instrument control
- Engineering and Scientific Computing
  - Data analysis
  - Simulation and modeling
  - Visualization
- Exploratory Computing
  - algorithm development
  - artificial intelligence programming, cellular automata, evolutionary programming

Forth Language

Stack Operations:

```
DUP  SWAP  ROT   DROP  OVER
>R   R>    ?DUP  NIP   TUCK
PICK .S    .     2DUP  ...
```

Examples:

```
                    →
                 2    2
1 2 .S           1    1
```

```
                 2    1
1 2 SWAP .S      1    2
```

```
                 3    1
                 2    3
1 2 3 ROT .S     1    2
```

Forth Language

Integer Arithmetic:

```
+    -     *      /    */
MOD  /MOD  1+     1-
NEGATE     ABS
```

Examples:

```
3 8 * . 24 ok

56 5 MOD . 1 ok
```

Forth Language

Relational Operators:

```
= < > <= >=
0= 0< ...
```

Examples:

```
1 3 < . -1 ok

4 0= .  0 ok

-5 -2 <= .  -1 ok
```

Forth Language

Bitwise Operators:

```
AND     OR      XOR  INVERT
LSHIFT  RSHIFT  2*   2/
```

Example:

```
: byte-swap ( n – m )
   DUP 8 RSHIFT SWAP 255 AND 8 LSHIFT OR ;

4096 byte-swap .  16 ok
```

Forth Language

Branching:

```
IF ... THEN
IF ... ELSE ... THEN
CASE ... OF ... ENDOF ... ENDCASE
```

Example:

```
: even?  ( n -- )
   2 MOD 0= IF ." YES" ELSE ." NO" THEN ;

5 even?  NO ok
8 even?  YES ok
```

Forth Language

Looping:

```
DO ... LOOP      ?DO ... LOOP
DO ... +LOOP     ?DO ... +LOOP
I  J
BEGIN ... AGAIN
BEGIN ... UNTIL
BEGIN ... WHILE ... REPEAT
```

Example:

```
: 2^  ( n – 2^n)  1 SWAP LSHIFT ;

: pow2-sum ( n – m | sum of terms 2^i, i=0,n-1)
0 SWAP 0 ?DO  i 2^ + LOOP ;

 10 pow2-sum .  1023 ok
```

Forth Language

Indefinite Loop Example:

```
: pad2 ( n – m | m is next power of 2, >= n)
DUP 0 <= IF DROP 1 THEN 1
BEGIN
  2DUP >
WHILE
  2*
  REPEAT
  NIP ;

348 pad2 . 512 ok
```

Forth Language

Recursion Example:

```
\ Find the greatest common divisor of two
\ integers

: gcd  ( n1 n2 -- gcd )
   ?DUP IF SWAP OVER MOD RECURSE THEN ;

1050 432 gcd .  6 ok
```

From A Beginner's Guide to Forth by J.V. Noble

Forth Resources

- Forth Programmers Handbook
- Forth Code Index
- comp.lang.forth

Forth in Python:

http://openbookproject.net/py4fun/forth/forth.html