FORTH

Eine ungewöhnliche Programmiersprache

Stephan Forth, DF6PA
OV Mainz, 14.04.2018

Fahrplan

1. Geschichtliches, wer, wann, was, warum
2. Definition, was ist FORTH
3. Einsatzgebiete
4. Eigenschaften
5. Wortkonzept
6. FORTH auf Microprozessoren
7. AMForth
8. Los gehts, ab hier wird rumgespielt
9. Links, Bücher, weitere Infos

Geschichtliches, Wer, Wann, Warum

- Erfinder Charles H. Moore geb. 1938
- Doktor der Physik 1960
- Arbeit im Bereich Astrophysik und Astronomie
- Autodidakt in der EDV
- Arbeitete zunächst mit LISP und Fortran 2

Geschichtliches, Wer, Wann, Warum

- Erste Entwicklung von FORTH 1968 auf einer IBM 1130 mit IBM 2250 Grafik Terminal
- Entwickelt zum steuern von Radioteleskopen und nur für den eigenen Gebrauch
- 1973 erste offizielle fertige Version
- Später CHIP Entwicklung, FORTH in Silikon NC4000, RTX2000 (Spaceshuttle)

IBM 1130

Geschichtliches, Wer, Wann, Warum

Name sollte FOURTH lauten, aber IBM 1130 lies nur 5 stellige Bezeichner zu!

1. Erste Generation (FIRST) Digitaleingabe einzelner Bits
2. Zweite Generation (SECOND) Assembler (Maschinensprache)
3. Dritte Generation (THIRD) Hochsprachen wie FORTRAN, LISP (nicht interaktiv)
4. Vierte (FOURTH) wurde dann FORTH

Was ist FORTH

- Interpreter
- Compiler
- Laufzeitsystem
- Entwicklungssystem

Einsatzgebiete von FORTH

- Weltraumfahrt, Steuerung von Satelitten und Teleskopen, Experimente im Spaceshuttle und der ISS
- Paket-Tracking von Federal Express
- Steuerung Münchner Teilchenbeschleuniger
- Messgeräte in der Autoindustrie
- Überwachungskameras der internationalen Atom Aufsicht
- OpenFirmware in SUN (jetzt Oracle) Maschinen und PowerMacs

Besondere Eigenschaften

- sehr vielseitige Sprache mit breitem Anwendungsspektrum
- nach den Vorlieben seines Erfinders schnell und kompakt, vorgesehen für Echtzeitanwendungen
- betriebssystemtauglich
- große Unterschiede zu Sprachen wie Pascal, Algol oder Basic
- Stackkonzept verhindert Entsprechung bestimmter Datenstrukturen
- lauffähig auf Maschinen unterschiedlichster Größe
- sehr geeignet für Meßdatenerfassung, Prozess- und Maschinensteuerung, automatische Testsysteme

FORTH verwendet Postfix (RPN/UPN) Notation

Prefix: + 3 4

Infix: 3 + 4

Postfix: 3 4 +

Dadurch keine Klammern notwenig, klassisches FORTH kennt keine Gleitkommazahlen, nur Integer!

Weitere Eigenschaften

- Strukturiert, also kein GOTO wie bei Basic oder Fortran IV
- Erweiterbar, jedes programmieren erweitert das System
- Sehr schnell, nur ca. 50-100% langsamer als entsprechende Assembler Routinen
- Wortkonzept lenkt automatisch zu Bottom-Up oder Top-Down Entwicklung
- Wenige Assembler Routinen, der Rest ist bereits FORTH
- Sehr gut portierbar

Top Down

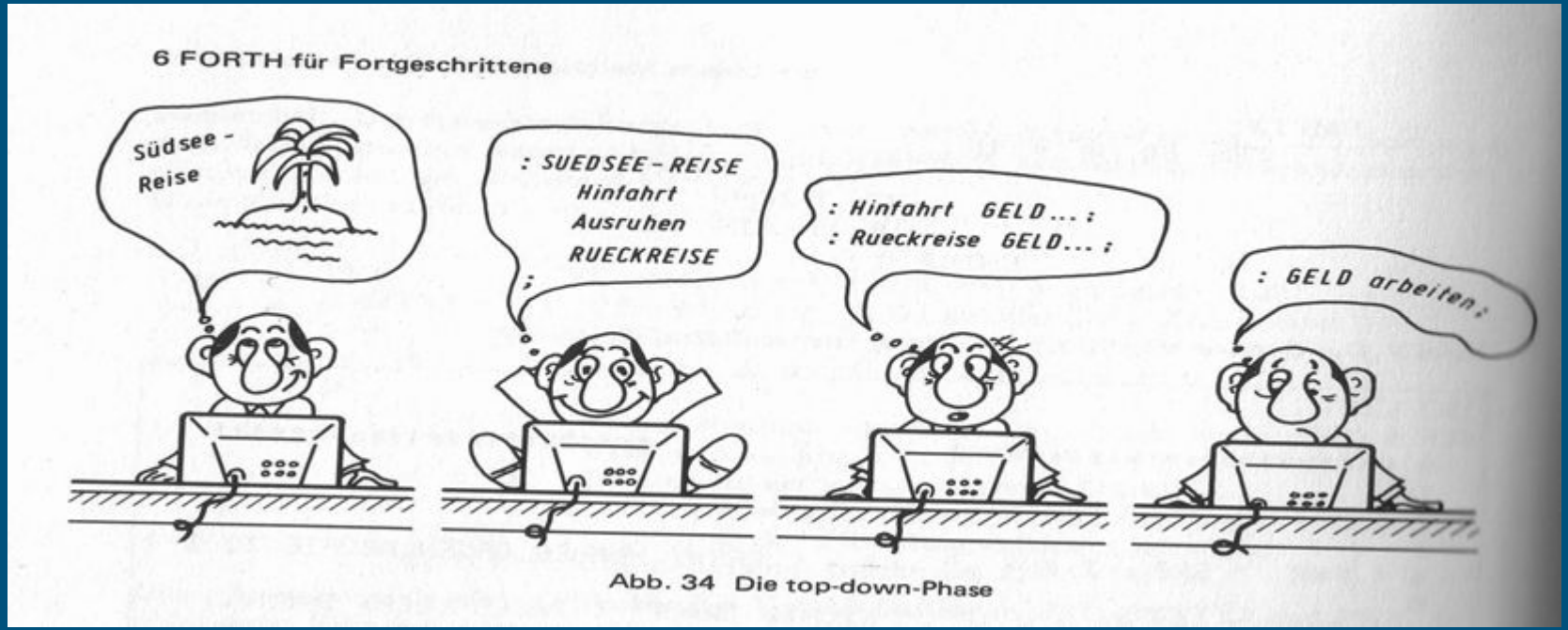


Abb. 34 Die top-down-Phase

Wortkonzept

- jedes Haupt- oder Unterprogramm, jede Subroutine ist ein Wort
- Wort String aus ASCII-Zeichen der Länge 1 – 31
- Wort setzt sich zusammen aus anderen Worten oder Maschinen- bzw Asm-Code
- erzeugt der User ein neues Wort, wird im 'Dictionary' ein neuer Bereich angelegt
  - Name in ASCII-Bytes
  - Codebereich mit Wortadressen bzw Maschinen-/Asm-Code

Beispiel

```
: Moin

.” Moin moin zusammen “

cr

cr

;
```

Details

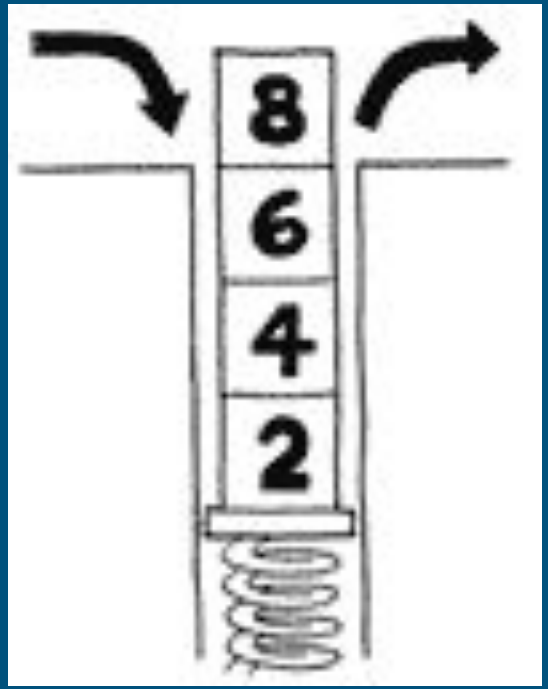
- FORTH-Programmierung ist stackorientiert
- im Gegensatz zu anderen (höheren) Sprachen ist die Arbeit mit dem Stack essentiell
- alle Notationen in UPN
- Programmierer ist dafür zuständig, dass der Operator die entsprechenden Daten auf dem Stack vorfindet (und auch in der richtigen Reihenfolge!)
- Ergebnis wird ebenfalls auf dem Stack abgelegt
- kein Typechecking o.ä. (alles Int)

Beispiel

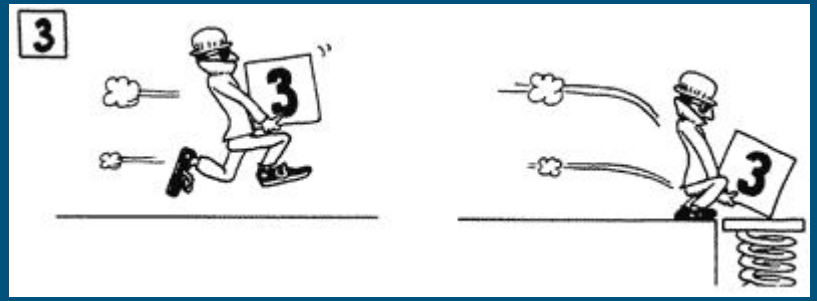
```
: addiere ( n1 n2 -- summe )
+
;

2 3 addiere

.

5 OK
```

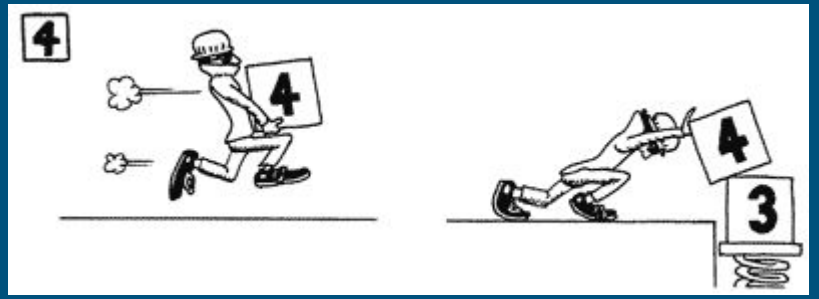
Stack

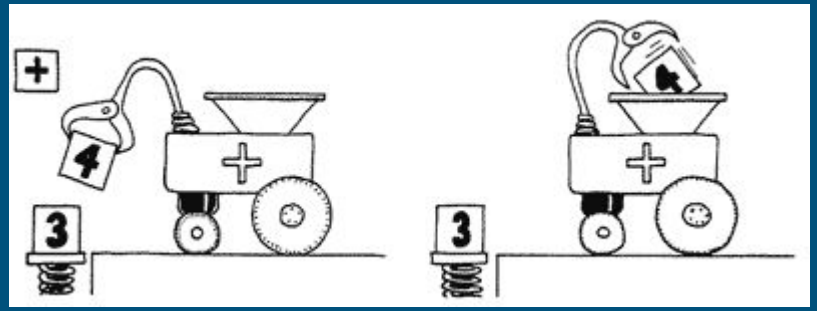
- First in, last out
- Zugriff nur sequentiell
- alles geht über den Stack

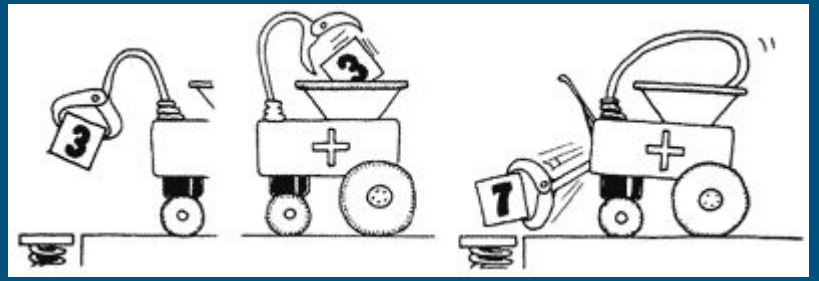


Addiere Beispiel









Stackmanipulation

```
DROP   (n1 -> )
DUP    (n1 -> n1,n1)
SWAP   (n1,n2 -> n2,n1)
OVER   (n1,n2 -> n1,n2,n1)
ROT    (n1,n2,n3 -> n2,n3,n1)
```

4.0 FORTH auf Microprozessoren

- Typisches FORTH ist sehr klein, z.T. kleiner als 4KB
- Inline programmieren möglich
- einfache Tests

5.0 FORTH auf Arduino

- AmForth
- FlashForth
- ESP Forth

AMForth

http://amforth.sourceforge.net/

Wird auf den Arduino “gebrannt” und ersetzt den Original Bootloader, programmieren mit der Arduino IDE danach nicht mehr möglich! Kann aber mittels Brenner rückgängig gemacht werden.

Besonderheiten von AMForth

- Kein Zeileneditor
- Kein Blockeditor
- Kein Forget (aber Marker: http://amforth.sourceforge.net/TG/recipes/Forget.html)
  (marker ist bereits aktiviert)
- Terminal sollte Zeilenorientiert sein (Python Script unter Linux, z.B. Termite.exe unter Windows
- Zur Not geht Putty, Screen, minicom, aber Achtung: kein Zeileneditor!

Ein paar Worte für den Anfang

.	gibt das oberste Element des Stacks aus
.” Text “	gibt den Text “Text” aus (auf Leerzeichen achten)
emit	(n1 --) gibt das entsprechende ASCII Zeichen aus (42=*)
spaces	(n1 --) gibt eine Anzahl Leerzeichen aus

Einfaches Beispiel

```
\ Druckt ein grosses F
\ geklaut bei FORTH INC

marker -testsf
( Grosses F )

    : star 42 emit ;
    : stars   0 do  star  loop ;
    : margin  cr 30 spaces ;
    : blip margin star ;
    : bar  margin 5 stars ;
    : GF    bar blip bar blip blip cr ;
```

Hinweise zum rumbasteln

Immer marker verwenden!

Möglichst Zeilenorientiertes Terminal verwenden!

Links

AmForth
http://amforth.sourceforge.net/

Floating Point Erweiterung (von mir ungetestet)
https://github.com/lnmaurer/amforth-float

FORTH Gesellschaft
https://www.forth-ev.de/

FORTH INC
https://www.forth.com/

Damit ich nicht immer suchen muss

amforth flashen:

```
sudo avrdude -c USBasp -p m328p -e -U flash:w:uno.hex:i -U eeprom:w:uno.eep.hex:i -U efuse:w:0xFF:m -U hfuse:w:0xD9:m -U lfuse:w:0xFF:m
```