

VolksForth Renovation: State Aug 2023

Philip Zembrod <pzembrod@gmail.com>
<https://github.com/forth-ev/VolksForth>

Overview

- Motivation & planned changes
- Fowler & the art of refactoring
 - Tests
 - Automation
- Sequence of change steps
- Platforms
 - C64/C16, MSDOS, CP/M, AtariST
- Merging forks
- Further plans

Motivation and planned changes


- Improved build platform for cc64
 - capable of stream sources
 - kernel as small as possible
- ANS

Planned changes:

- Implement stream source handling
- Make block words optional
- Merge again the forked code bases of C64/C16, AtariST, CP/M, MSDOS
- Migrate unified code base towards ANS

Safety net for changes

- Hayes tester + ANS test suite

- Hacky ANS adaptation via 100 lines of shim code 
- Comment out tests where needed
- Relevant tests: Core, CorePlus, CoreExt, Double, Block

- Log file output

- Diff against golden files

```
: cells 2* ;
: s" [compile] " compile count ;
  immediate restrict
: c" [compile] " ;
  immediate restrict
: [char] [compile] ascii ;
  immediate
: char [compile] ascii ;
: 2>r r> -rot swap >r >r >r ;
: 2r> r> r> r> swap rot >r ;
: 2r@ r> r> r> 2dup
  >r >r swap rot >r ;
: again [compile] repeat ;
  immediate restrict
: compile, , ;
: defer! >body ! ;
: defer@ >body @ ;
```

Automation makes code malleable

- Gnu-Make
 - + powerful flexible rules
 - - cryptic syntax
- emulators ...
 - VICE, x16emu, dosbox, RunCPM (, Hatari)
- ... made scriptable with tricks
 - If guest code deletes file NOTDONE, then host code terminates VICE &. x16emu
 - Added options -i and -o to RunCPM
- ascii2petscii, petscii2ascii, unix2dos, dos2unix
- target compile make rules
 - with log file to check build success

Boundary conditions

- tests as safety net for all changes
- Hayes & ANS tests are stream sources
- stream INCLUDE uses TIB.
- original TIB length: 80 char
- tests beyond Preliminary and Core are longer than 80 char
- automatic verification of test and target compile runs

Order of steps per platform (all tests & compiles via make)

- `.(Hello World)` in emulator
- log file output
- implement `INCLUDE.fb`
 - with helloworld test
- tests: `preliminary.fth` & `core.fth`
- target compile from `kernel.fb`
 - tests: `preliminary` & `core`
- convert `kernel.fb` -> `kernel.fs`
 - initially one file
 - split into smaller files later
- migrate target compile `.fb` -> `.fs`
 - stepwise starting with `loadscreen`
 - tests: `preliminary` & `core`
- enlarge TIB to 126 char
- remaining tests
 - `coreplus`, `coreext`, `double`, `block`
- integrate `INCLUDE` in `kernel.fs`
 - tests: all
- extract block words from kernel
 - some words become deferred
 - tests: all respectively all except `block`

C64/C16

- first platform
 - first `INCLUDE`
 - “steps per platform” sequence is established
 - `ans-shim.fth`
 - 314 of 3431 lines of test code commented out via `\vf`
- emulator VICE
 - input via `--keybuf`
 - terminated by host, when guest signals by deleting a certain file
 - ASCII-PETSCII
- all target compiles on C64 (x64 is the fastest VICE)
 - self-hosted C16 target compile exists as proof-of-concept

Detour Commander X16

- low-hanging fruit due to similarity to C64
 - shared code base - hence 6502, not 65C02
- emulator x16emu
 - input via `-bas` (“type in Basic program”)
 - terminated by host, like VICE
- no block words
- cc64 is first self-hosted C compiler on X16

MSDOS

- emulator dosbox
 - input via `-c <command1> -c <command2> ...`
 - terminated via `-c exit`
- INCLUDE exists for .fb files
 - file type detection for block and stream files
- second pass through “steps per platform” sequence
 - more code reshuffling needed to extract block words than for C64

CP/M

- emulator RunCPM
 - enhanced: flags for piping console I/O to/from files
 - terminated via `exit`-command
- done:
 - `INCLUDE.fb`
 - preliminary and core test via `make`
- next:
 - enable nested `INCLUDE`
 - target compile

AtariST

- next in line
- emulator Hatari
 - good press re scriptability :-)

Merging of platform forks (planned)

- extract Forth code to `common/`
- keep code words out of `common/`
 - too many `(6502 (8080 (8086 (68k)` would be messy
- open question: how to avoid collisions like
 `code dup ... end-code`
and
 `: dup ... ;`
- ideas:
 `(?ifndef dup : dup ... ; ?)`
 `? : dup ... ;`

x86-Linux

- planned after merging of code
- great as target compiler platform
- great as learning tool
- writing of ELF binary already researched

Thank you!

Any questions?