Entry into the state-FORTH#

Table of Contents

So you can begin immediately, is described in this chapter,

- How to start the system
- How to find his way in the system
- How to create a finished application program
- How to build its own working system

The surface #

When you start VOLKS4TH of the DOS level, logs volksFORTH83 a Einschaltmeldung that the version rev. <xxxx> contains.

What you see now from state-FORTH, the surface of the interpreter. FORTH systems and thus are almost always socially FORTH Interactive Systems, where you can check just one line of thought developed and realized immediately.

MS-DOS / ATARI PORTFOLIO

The most striking thing about the state-FORTH-surface is the inverse status cell in the bottom screen line, which can be with **status off** off and on with **status on** again.

This status bar shows from left to right following information, where "/" for "or" position,

<2/8/10/16>	the currently valid number basis (decimal)
S <xx>	indicates the number of values, the readily available for processing
Dic <xxxx>	is called the free space
Scr <xx>	is the number of the current code block
: A / C:	is the current drive (MS-DOS)
<name>. <ext>	Displays the name of the file being edited. Files in MS-DOS both a name and a three-letter identifier <name>, <ext> the extension, including files can be created without extension.

FORTH FORTH FORTH	displays the current search order according to the vocabulary concept. An example is the assembly language commands: These are located in a vocabulary called **ASSEMBLER assembler words** and shows them the instruction set of the assembler. Please pay attention to the right side of the status line. where now see **assembler forth forth** is. But since you now - still - do not want to use assembly language instructions, please switch with **forth** the search priority to go back. The status line shows again the familiar **forth forth** forth.

For orientation in the working system is the state-standards-FORTH some words are available:

Words	command shows you the list of FORTH, the available words. The list stops at a key press with the issue or comes to an ESC.
Files	**MS-DOS**: shows all created in the system logical file variables that are associated handle numbers, date and time of last access and its corresponding physical DOS files. Such FORTH file is signed only by the mention of her name. The MSDOS files in the directory are displayed with **dir**.
Path	**MS-DOS** to date: a full Pfadunterstützung after MSDOS principle, but completely independent of it. If no path set. so there **path** nothing.
Order	describes the search in the command lists (vocabulary).
Vocs	call all these subdirectories (vocabularies).

Working with program and data files (MS-DOS / ATARI ST / CP / M)#

To write any program text (source text), you need a file that records the program texts. This file must be registered for processing.

FORTH folk goes to the system start from the first file VOLKS4TH.SYS as the current file. VOLKS4TH.SYS is the control file from which your FORTH system was built, so you declare the file. you want to edit with this command sequence:

(((use <name>. <ext>)))

As an example, the file is registered with **test.scr users test.scr**.

Would you like to use, however, a completely new file for your programs. <name> you think about a name, an ID <extension> and a reasonable size in kilobytes. Then enter:

(((makefile <name>. <ext> <Größenangabe> more)))

Then the file is created on the drive and registered to edit. A **use** is then no longer necessary.

The Editor #

Atari ST, MS-DOS, CP / M

To edit text blocks contain mostly older Porth systems editors, who deserves the name more than a tent when you add the bits individually, with hole punch and tape reader was required to notify the hardware. In contrast, offered an editor with which you can edit each a whole cell, certainly some comfort. Since, however, with such a line editor today make more than a state and certainly can not compete with other languages, you can work in the people-FORTH course with a comfortable full-screen editor.

In this editor you can edit two files simultaneously:

A foreground file, and the current **isfile** Hintergrundfile **fromfile**. Therefore, the editor two file names are displayed. The word **use** logs to a file automatically as well as a **isfile fromfile**, so that shift and copy operations relate only to this one file.

In the Editor Always a FORTH-screen - that is 1024 bytes - In the normal distribution in 16 cells, each with 64 columns is presented.

There is a character and a cell memory. This allows characters or cells within a screen or copy, move, or even between two screens. This will prevent accidentally lost text, can not run in the functions where this character to the bottom or side, are inserted from the screen.

HELP and VIEW #

The editor supports the 'Shadow-concept'. For each text screen, there is a comment screen. This increases the readability of FORTH programs much faster, you know, is good FORTH-style self-documenting! At the touch of a button CRTL-F9 the editor to comment screen available. Thus, comments 'are identical' made to the source texts. Shadow This concept is also used at the word **help** that prints to a word an explanatory text. This word is so used:

(((help <name>)))

HELP only shows correctly on the statement, if such a text is to screen a shadow exists.

Often, one might also see the definition of a word, for example Stack read the comment or the exact way of working. There are command

(((view <name>)))

Thus the screen - and of course the file is called - has been defined on the <name>. This procedure replaces (almost) a decompiler, because of course it is much more convenient and you, too, all source texts of the system are available.

Surf to please **view u?** A look at the word that you output the contents of a variable. Use

(((fix <name>)))

(Fix - engl. = Error), so also the editor is enabled, so you can immediately make specific changes in the source text. In the editor you are first prompted for a command "Enter your ID:", the answer simply by pressing the button <cr>.

Now you are editing in the editor with which you, the source blocks. The cursor is placed after the found word **u?** . Now you can scroll with PgUp or PgDn to return with <ESC> screens or from the editor to FORTH.

Of course, for **view**, **fix help** and the corresponding files on the drives 'at hand', otherwise an error message. The VIEW function is also available within the editor, you can then request the keystroke CTRL-F on the right of the cursor word. This is especially useful if you want to take a definition from another file or not sure how the word is a stack comment.

Open and edit a flow #

Finally to create and modify source texts for programs - a beautiful definition of editing. You have a file **test.scr** determined as described above with

(((makefile test.scr 6 more)))

created. You have a file called **TEST.SCR** with a length of six blocks (6144 bytes = 6KB), consisting of the screens 0-5, of which the numbers 0-2 for source text, the other destined for comments.

In order to reach the editor, there are three possibilities: (((<screen#> edit))) or (((<scr#> l)))

calls on the screen with the number <scr#>. If you have already edited. calls

(((v))) recently edited the screen again. This is the last edited or, and this is very helpful, the one that caused an abort when compiling. folk FORTH breaks - like most modern systems - a program error during the compile and marked from the spot where the error occurred. Then you need only enter **v**. The faulty screen is loaded into the editor and the cursor is placed after the word that caused the demolition of the build process.

As an example of a block be edited:

(((1 edit))) You will be asked to enter the three-letter abbreviation. The shortcut is then entered in the top right of the screen and gives the "programmer's id again. If you want to type anything, you just press <cr>. To use these blocks, there are some agreements, of which I will mention two:

1. # 0, the block number is never! for program texts used - there are usually found explanations and instructions on the program and author.
2. In the cell 0 of a block is always entered a comment that starts with \ (skip line). This backslash ensures that the following line is ignored as a comment. In this cell 0 to write there the names of words that are defined in the block.

Switches of the Editors #

MS-DOS

When editing, the following functions:

	Key		function
F1	gives assistance to the Editor		
ESC	leaves the editor with the immediate saving the changes.		
CTRL-U	(undo) makes all changes, which have not yet written back to disk.		
CTRL-E	leaves the editor without an immediate saving.		

CTRL-F	(fix) is looking for the word to the right of the cursor without leaving the editor.
CTLR-L	(showioad) loads the screen from the cursor position.
CTRL-N	inserts at cursor position a new cell.
CTRL-PgDn	splits within a cell that line.
CTRL-S	(Scr # list) from the number of screens being edited on the stack: eg for a load or following plist.
CTRL-Y	deletes the cell at cursor position
CTRL-PgUp	fits within a cell, the lower right part of the cell to the top line a (join).
TAB	moves the cursor in front of a large tab.
SHIFT-TAB	moves the cursor back a little tab.
F2	search and replace (/) expects a string to be searched, and a string to be inserted instead. If a match is found, can be replaced by R (replace) the search string, abort with <cr> the search or search with any other key to the next match. In this way, can the source texts can also search for a string. As a replacement string is then entered <CR>.
F3	bring a cell in the cell buffer and deletes it in the Screen.
F5	bring a copy of a cell in the row buffer.
F7	inserts the cell from the cell buffer in the screen.
F4	F3 as, only for a single character.
F6	F5 like, but for a single character.

F8	F7 corresponds, based on a character.
F9	exchanges the current file (**isfile**) with the background file (**fromfile**). Press F9 again swapped again referring to the old state is restored. This function is useful when you edit a file that can be in between but with CRTL-F to display a word. It turns F9 (= fswap) restore the old file distribution, which has been amended by the **fix**.
SHIFT-F9	switches to the comment text (shadow screens) and at the next press back.
F10	puts the current screen briefly aside to him to work on with a press of F9 again. If you refuse the copy features a state-FORTH **copy** or **convey** with the message*TARGET BLOCK NOT EMPTY*because **isfile** and **fromfile** are different, F10 turn causes confusion.

Another note to the editor:

The editor supports only the copying of cells. You can copy this kind screens, but sometimes necessary when inserting screens in the middle of a file is a bit cumbersome. To copy the entire screen within a file or a file to another are the words used in folk FORTH **COPY** and **CONVEY**,

Example: CLS #

For your first program you now please take the following text in a screen 1:

(((\ CLS clears the screen : Cls (-) full page,)))

Now press SHIFT-F9 and enter a comment in this screen, the Declaration on **CLS** in the same cell, such as the definition of the source text, eg:

(((\ \ CLS

CLS clears the screen by the words full (screen window to full size) and PAGE (figure- Clear screen window) recourse.)))

A SHIFT-F9 again to take you back to the source text (source code back). To get this explanatory text appear after compiling with **help cls**.

If you scribbled a block, scroll with PgUp or PgDn back before the next block. Are you satisfied with your program, press ESC, and then the screen changed immediately saved.

Then you'll notice something: The screen no longer works correctly, it will remain above lines, not the scroll,

That's right, so the last used source text does not scroll away to the top. In order not to lose another two rows of the screen has the window in which you are working, no frame. How to work with frames and windows, is shown in the editor, which is available as all other system components in the source text.

In order to rapidly liberate the future from the unfortunate situation with the reduced screen size, use your first self-written program **cls**. After all, the entry of **full** you again the entire screen available, whereas **page** deletes only the current window.

In compiling volks4TH - as in most FORTH systems - is performed on <scr#> load: (((A load)))

compile your word **CLS** with unimaginable for 'C' or Pascal programmers speed into FORTH. In order for your mini-program is now ready for execution. At the same time you get the message that a word called **CLS** already exists: CLS exists. This only has the consequence that, after a redefinition of the old word of the same name can not be accessed. reported a possibility of this same name with an already existing word, would be to use a vocabulary.

Enter once **words** one, then you will notice that your new word in the dictionary is **CLS** top (press the <ESC> tests to stop the issuance of **words** or any other key to stop it),

To verify the results of your test first programming, type now:

(((cls)))

And lo and behold, the entire screen is dark. It would be nice if your program would end its work with a message. To change this, please take away the old word once **CLS**:

(((forget cls)))

Please check with **words** whether **CLS** was really forgotten. Then call again on the editor with **v**. Now use the word for the beginning of a string **."**, the word for your end **"** and the word for a line feed **cr**. Your screen will look like this one:

(((\ CLS clears the screen with a message : CLS (-) full page . 'Screen properly cleared! " cr;)))

After **."** must be a space. FORTH by default because the space used as a separator between individual words, so this does not count with spaces to the string (string). Then you leave the editor and compile your program, such as start and had it. The change is proving to be successful, and you learned how easy it is in FORTH writing, testing and modifying parts of the program.

A great help for both the programmer and for the future users is information about what is being charged and what has been already compiled. Add

(((cr. (function) installed)))

one has to be returned during the loading of these declarations. The word **.(** introduces a comment in the interpreter, the closing bracket ends **) cr** him and of course for the new line, the carriage return, responsible.

Build in editor - Showload #

A special feature of state-FORTH, is that even in the editor functions of the interpreter / compiler is available. This interpretation and compilation in the editor called Showload.

CTRL-F	(fix) is looking for and displays the word to the right of the cursor without leaving the editor.
CTRL-L	(showload) loads the screen from the cursor position.

To see. what makes this Showload function, please enter now following screen:

(((\ A test for the showload : Inc + 1; : Dec 1 -; \ \

15 inc. 15 dec inc. 15 15 + 2 spaces.)))

This screen will now be compiled and interpreted in the editor!

In addition, please put the cursor by pressing Ctrl-Home in the first line on the sign (pre) \ (pre) (skip line). Now press CTRL-L to load the screen. nationally compiled FORTH now up to the line that begins with (pre) \ \ (pre) (skip screen). The words and **inc dec** the system are now known and can be used. Next, move the cursor after the (pre) \ \ (pre) and press CTRL-L to further interpret the editor. Now you see the issues at the appropriate point appear in the editor. Here, the content of the screen intact, of course - you leave the editor with ESC and check the contents of the screens with **<scr#> list** on and you see only the instructions, but no more spending on.

A typical application of this would be **showload** recompiling a word for a small change in the interactive or adding words. you just need.

Creating an application #

MS-DOS / ATARI ST / CP / M / ATARI 8Bit

You want to save your 'program' is now a standalone application. To do this, first expand it a little bit. Add now the following definition in a new cell added:

(((: Runalone cls bye;)))

RUNALONE CLS First executes and returns to the operating system. Now, compile again. You run **RUNALONE** but not, or you would leave FORTH yes (**bye**).

The problem is rather to save the system so that it immediately after loading **RUNALONE** running and nothing else. volksFORTH833 is prepared in two places for such purposes. In the words and are **COLD RESTART** two 'deferred words' named **'COLD 'RESTART** or who do not normally, but the user can be changed later.

Use here **'COLD** to also have to stop the startup message. This means including a

((('Is runalone' cold)))

and save the whole thing with **savesystem cls.com** back to disc. You have created your first application you can call from the operating system level with **CLS**!

A little disappointing but it is already. The supposedly compact FORTH About 20KByte needed. To execute such a ridiculous function? There is something wrong yet. Of course, yes there a number of system components with were saved from Fileinterface the assembler, the editor, etc., for our program is not needed at all.

To solve this and similar problems, there are the file KERNEL.COM. This program contains only the kernel and the Fileinterface, and reflects the shared library (runtime library) of other languages. **MS-DOS**: So you shop KERNEL.COM and compile your application with **include multi.vid include test.scr**

The previous load **MULTI.VID** is necessary because FORTH systems rarely have left; **MULTI.VID** is not loaded before, then the FORTH system **full** the word is unknown. Then as usual in **'COLD RUNALONE** about the system and save back to disk. From the MS-DOS can level off this program generation with (((kernel include multit.vid include test.scr)))

Perform both of these cells ((('Is runalone' cold savesystem dark.com)))

the final instructions on your screen are. Thus the FORTH interpreter is instructed to save the completed application **DARK.COM** to disc.

You now have available a relatively compact version. Of course, this could also have reduced significantly, but you would need a compiler target, which you could assemble only the most required system components selectively from the source text. The described method can also build larger programs and as stand-alone applications save.

Creating your own FORTH system #

MS-DOS / CP / M

This file VOLKS4TH.COM is intended as a working version.

If it contains all the important system components such as editor, printer interface, tools, Decompiler, Tracer, etc. you do not like the compilation, you can always put together a system tailored to your specific needs.

Key to the Loadscreen of the file is **VOLKS4TH.SYS**. You can then parts of the system, you do not need wegkommentieren with the backslash (pre) \ (/ pre) or delete the corresponding cells completely. You may, of course, the Loadscreen add your own files. Conforms to Loadscreen your wishes, save him back with <ESC>, and leave the system with **BYE**. Now load the file **KERNEL.COM**. Then type, (((include volks4TH.sys)))

Once the system finished compiling. writes the statement (((savesystem volks4th.com)))

of load screens, a new version of this disc is overwritten volks4TH.com on your old file (backup copy!), so you get the next load volks4TH.com your own system.

Of course you can 'save your' system under a different name. You can also change system settings. Our working version works - default - now in the decimal number system. Of course you can change with **hex** to hexadecimal, and we think this is much more useful because testify at all memory addresses in the decimal little something (or do you know if memory parts is not in the screen memory or 978 584?). Would you immediately after the shop to work in hexadecimal, you can save with this **SAVESYSTEM** by typing the command FORTH cell **savesystem volks4TH.com**.

In addition, we recommend to all figures on nine urgent use of the so-called prefixes,

	Prefix		Number System
$	For hexadecimal,		
&	For decimal and		
%	For binary numbers.		

It is reported so that any files are not compiled - or even worse, wrong - because you're just another number in the system. Moreover, it is possible to combine hexadecimal and decimal numbers as desired. Depending on what makes more sense now. In the source texts, see enough examples of this.

Particularly fast and comfortable state-FORTH operates, of course, if all parts of the system are stored on a hard drive. You should ensure that it establishes its own directory and set **PATH** and **DIR** accordingly.

Also working with a RAM disk is possible in principle, but not highly recommended. FORTH is very machine-and system crashes, therefore, before the start not completely ruled out.

The printing of the source texts of the system is certainly useful to see examples of how to deal with voiksFORTH83. Shall place e.g. the screen editor and command-editor complete applications is available in the source text.

Flies in which individuals are on your disks and whether they contain this screen is, **README.DOC** in the file. First you need to Merge the printer interface, if it is not already there. Responding to the input of your volksFORTH83 **PRINTER ?**, with a printer so the interface is not available.

Printing Screens #

MS-DOS / ATARI ST / CP / M

Should be present in your system no printer interface, so you download it from the command line with-FORTH

(((include <printername> prn.)))

after. If your printer does not appear in the list. you use instead **graphic.prn** or **epson.prn**. Most printers can work as an IBM Graphic Printer or EPSON FX / LX printer.

The Printer Interface few words about the issue of a formatted listings are included. **PTHRU** Prints a range of screens, each with 6 screens in on a DIN A4 page compressed document. Similarly, the word **DOCUMENT** works, but is this a code word next screen, the corresponding Shadow Screen printed. **LISTING** Print an entire file in such a way to get such a concise listing of a file with detailed comments.

Glossary #

	Word		stack comment		Description
Pthru	(up -)	always prints the specified blocks to six on a page.			
Document	(from working to -)	**pthru** as prints, but three blocks and three comment blocks			

		of code on a page
Listing	(-)	created a listing of the entire file by three code blocks, and three comment blocks are printed on one page
Plist	(scr # -)	print out a block on the specified printer
Scr	(- addr)	is a variable that contains the number of screens being edited. Compare (pre) r #, list, (error (/ pre)
R #	(- addr)	is a variable that contains the distance of the label being edited from the beginning of the currently edited screens.

Printer adjustment #

The printers work adapting the system in the case VOLKS4TH.SYS by the line (((include <printer> prn.)))

made. In this adaptation are - in addition to the pure output routines - contain a number of useful words, with which the printer controller can be made very comfortable.

In the working system, the printer interface is included. Need to make changes, you can change with the editor of the Loadscreen VOLKS4TH.SYS and compose a new work system with: (((kernel include volks4TH.sys)))

Of course you can also use the Loadscreen in its current form and the Printer Interface every time 'by hand' with (((include <printer>. prn))) recharge.

Unfortunately at the moment still in the people-FORTH German and error messages are mixed and the engllsche **heip** that reveals an explanatory text only if it is present.

Overview #

The most important commands again at a glance:

Status	status controls the cell
Words	displays the currently available commands

Files	displays the registered files
Path	modified or call the file search path
Order	search order of the command lists the groups on
Vocs	lists all available command groups
View	displays and
Fix	edited the text a specific word
Help	shows - if any - the comment text of a word
Full	the Blidschirmfenster switches to full size
Page	deletes the current window / screen
Index	non - resident - the contents of a block file shows
List	displays the contents of a screen.
Include	command loads a whole group or part of the program