

PREFACE TO THE THIRD EDITION

It is almost eight years since F83 was first released by Mike Perry and Henry Laxen. It has been widely distributed by many shareware and freeware distributors, as well as through many bulletin boards. It also has found its way into many real applications and useful products. Although we have seen many better public domain Forth systems brought out over the years, F83 still stands out because of its high quality and because it is available on three very popular microprocessors: 8080, 8086, and 68000.

The quality of F83 is testified by the fact that over the years, we have found only one bug in the 8086 F83 system (DOS Version 2.10). This bug was discovered by Mike Yantis at Maxtor Corp. The ENTRY cell in the user area contains 90H (NOP) and E9H (JMP) when the task is asleep. E9H causes a jump to the next task, thus skipping the current task. These two bytes are changed to CDH and 80H (INT 80) when the task is to be waken. INT 80H wakes up this task when the CPU control is passed to the task. This scheme works fine in most instances. This mechanism falls apart only if the waking up routine is activated by an interrupt, and if the interrupt hits when the CPU just finishes executing the NOP (90H) instruction and is ready to execute the JMP (E9H) instruction. Unfortunately, the waking up routine secretly changed E9H to 80H, whose behavior at this point is unpredictable and in most cases crashed the CPU. The probability of this occurring is very small, only about once in 100,000 interrupts, which were enough to bother Mike Yantis. Mike fixed this bug by choosing INT E9H to start the wakened tasks.

Discussing this bug in detail is meant to be a compliment to Mike Perry and Henry Laxen in their efforts producing the F83 system. It took a bug of such a oblique nature to escape Mike and Henry's tight grips.

I take this opportunity to revise the book and produce it using a laser printer. I am always amazed at how a laser printer can transform lies into truth. In spite of the laser, I like to give special thanks to Jay McKnight in reviewing the text and corrected many of my grammatical and technical errors.

F83 is one among the very few Forth systems which are useful while still understandable. *Inside F83* had helped many people gain the privilege to peek inside a fully functional Forth system. I hope it will help you also. Not just take a peek, but use it as a key and open a whole new field to yourself.

C. H. Ting

June 1991
San Mateo, California

PREFACE TO THE SECOND EDITION

After I implemented my first Forth system on a Data General Nova computer and got the 'OK' message, I went back home and told my wife: "I just promoted myself from an applications programmer to a system programmer!" I was so excited and my brain was so completely filled with the intricate details of the Forth fabric that the only way to get hold of myself was to dump everything on paper. That was the *Systems Guide to figForth*. I xeroxed it and brought a boxful to the then Northern California FIG meeting and it was sold out immediately. Apparently I had struck a chord in the Forth community which was desperately in need of documentation and instruction on Forth internals.

I was fortunate that a polyForth on LSI-11 computer was available at work. I tried to convince Forth, Inc. to publish a similar book on polyForth and obtained some support to proceed. I sent a draft manual, titled *Systems Guide to polyForth* to Forth, Inc. Somehow, Forth, Inc. decided not to publish or promote it, and had left it on their bookshelf. I heard that it found its way into the underground Forth circle in Southern California. PolyForth is concise and powerful, and it deserves better system documentation than what is provided. I was very impressed as I went through it screen by screen. I was delighted in picking a great mind, that of Chuck Moore himself. It was like poetry.

When Mike Perry and Henry Laxen released their public domain F83 system, I bought a listing to read. It was a very worthy product, with lots of tools and utilities. The best part is that it is complete with on-line documentation in the form of shadow screens. I thought there was little for me to contribute. As F83 was spreading wider, we started to hear more complaints about the difficulties in learning and using it. I reached a conclusion that Forth screens are good medium for programming, but a screen is too small a window for viewing and learning a large Forth system, even with shadows. In reading the source code, it is necessary to look at many screens at the same time, quickly moving from one screen to another while keeping everything in plain view. We have all been conditioned to read things in the printed form, making the best use of our visual system with instant zooming and panning capability. The visual system is very difficult to emulate with a 24 by 80 character screen.

Then Wil Baden came to one of the FIG meetings and showed the completely sorted index of F83 words in all vocabularies. I rushed to the front table and grabbed a copy of his handout with the index, which was the tool I needed to navigate through the F83 system. With the help of the index, lots of midnight oil, and ignoring my wife's orders to clean up my room, I was able to rearrange the source code of F83 in a form more tangible to mortal souls. Most of the work was simply rearranging the source code from the horizontal to the vertical format and fill the right hand side of the page with words taken from the shadow screens. I collected related source code and present it in a logical sequence, which often does not coincide with the loading sequence of the source screens. Once the code is ordered logically, you will find it is much easier to comprehend this very large and seemingly intimidating system.

F83 provides a very extensive and solid foundation for professional programmers to build application packages. It is also a very useful source for beginners to learn Forth programming style and techniques. Its problem, as in any large Forth system, is the fragmentation of functions in a multitude of words.

With more than 1000 words, it is very difficult to have a firm handle on

F83. However, functions a user needs to program a computer application or to use a computer application are not that many. Once you are familiar with those top level utility words, you can dig into the underlying low level words and use them to build your own castles. This book, I hope, will serve the purpose of showing you the power and the beauty behind the Forth language.

We are all indebted to Mike Perry and Henry Laxen for releasing the F83 system into the public domain. It certainly sets a higher standard for commercial Forth systems and forces Forth vendors to provide more powerful systems and better user support. Anything less than F83 will not be acceptable anymore. Thanks are also due to Dr. S. Y. Tang and Mr. John Peters for reading the manuscript and making numerous suggestions and corrections. The Chinese brush painting on the covers was provided by my mother, Mrs. I-Jean Hwang Ting. My father, Mr. C. W. Ting, was the editor and also managed the production of this book. This is a traditional Chinese family business, small but efficient and very Forth-like.

C. H. Ting

May 1985
San Mateo, California

INSIDE F83

CONTENTS

Part 1. Introduction to F83 system

1.	The heritage of F83 system	1
1.1	The roots of the F83 system	1
1.2	Advancements in Forth-83 Standard	3
1.3	Creators of F83 system	4
1.4	Features of F83 system	5
2.	Browsing F83 system	7
2.1	Listing the word names	7
2.2	Vocabulary	9
2.3	Viewing source code of word definitions	14
2.4	Shadow screen documentation	15
2.5	Files in F83	15
2.6	Printing utility	18
2.7	Debugger	18
3.	Using the F83 system	20
3.1	Create your own file	20
3.2	The editor	21
3.3	Loading and testing your program	24
3.4	Memory dump	24
3.5	Debugging your program	25
3.6	The 8086 assembler	27
3.7	Multitasker	31
3.8	Save a system image	32
3.9	The meta-compiler	33

Part 2. The Forth kernel

4.	Interface to the host computer	35
4.1	Virtual Forth computer	35
4.2	Forth computer hosted on 8086	36
4.3	Inner interpreters	40
4.4	Interpreters for in-line data and strings	44
4.5	Interpreters for control structures	45
5.	The Forth nucleus	48
5.1	8086 assembly language in Forth	48
5.2	Code definitions in Forth nucleus	49
5.3	Examples of code definitions	50

6.	Terminal input and output	52
6.1	The BDOS I/O calls to the operating system	52
6.2	Terminal output commands	53
6.3	Interpreting control characters	54
6.4	More sophisticated input commands	55
6.5	String commands	57

7.	The virtual memory	59
7.1	Mass storage and virtual memory	59
7.2	Disk buffers	60
7.3	The file control block (FCB)	62
7.4	Read and write disk files	63
7.5	Disk buffer management	64
7.6	Saving disk buffers to disk files	69
8.	Dictionary and vocabulary	71
8.1	Threading of the dictionary	71
8.2	Hashing and searching the dictionary	76
9.	Number input and output	80
9.1	Representation of numeric data	80
9.2	Input number conversion	81
9.3	Output number conversion	85
9.4	Double integer output	87
10.	Word parsing	88
10.1	Text processing	88
10.2	Input stream and input buffers	88
10.3	Low level parsing commands	89
10.4	High level parsing commands	92
10.5	String commands defined using PARSE	93
10.6	End-of-buffer condition	93
11.	Text interpreter	94
11.1	The operating system of Forth	94
11.2	Entering the text interpreter	94
11.3	INTERPRET	95
11.4	DONE? and X	96
12.	Compiler	98
12.1	The colon definition	98
12.2	Colon and semicolon	99
12.3	The compiler loop	100
12.4	Low level supporting commands	102
12.5	Immediate commands	103
13.	Structures in colon definitions	105
13.1	Compiler directives	105
13.2	Compiling numeric data structures	105
13.3	Compiling string literals	107
13.4	Compiling control structures	111
13.5	Address calculation for control structures	111
13.6	Control structure compiler directives	112

Part 3. Utilities in F83 system

14.	The CP/M-DOS files	115
14.1	CP/M-DOS file primitive commands	115
14.2	The file control block	116
14.3	High level file commands	117

14.4	Save core image to a file	118
14.5	Directory accessing	119
14.6	System level file commands	120
15.	Text editors	123
15.1	String utility	123
15.2	Terminal dependent deferred words	125
15.3	The cursor commands	126
15.4	Editing buffers	128
15.5	Line editing commands	131
15.6	String editor commands	132
15.7	Screen editor	134
15.8	The screen display commands	135
15.9	The screen editor commands	138
15.10	Configuring the terminal	139
16.	Viewing source screens	141
16.1	The view field	141
16.2	The view files	143
16.3	The viewing command	143
17.	WORDS	145
17.1	Output formatting commands	145
17.2	WORDS	145
18.	Disk file utility	147
18.1	Displaying screens in a file	147
18.2	Disk buffers	148
18.3	Single block copying	149
18.4	Multiple block copying	149
18.5	Multiple file copying	151
19.	Memory dump	153
19.1	The dumb DUMP	153
19.2	The smart DUMP	153
20.	Decompiler	156
20.1	Positional case defining word	156
20.2	Associative defining word	157
20.3	Decoding different classes of words	158
20.4	Sorting and execution tables	160
20.5	Decompiling different word classes	162
20.6	Word classification	162
20.7	The decompiler SEE	163
21.	Printing utility	165
21.1	Variables and setup	165

	10	
21.2	Printing two screens side by side	166
21.3	Printing 6 screens on a page	167
21.4	SHOW	169

Part 4. 8086 Specific utilities

22.	Debugger	171
22.1	Low level supporting words	171
22.2	High level trace commands	173
23.	Multitasker	175
23.1	Multitasking	175
23.2	User variables and the user area	175
23.3	PAUSE and RESTART	177
23.4	The multitasker	179
23.5	Task definition	180
23.6	Background tasks	181
24.	8086 Assembler	183
24.1	Assembly tools	183
24.2	8086 register definitions	185
24.3	Addressing mode operators	187
24.4	Defining words to generate opcodes	192
24.5	Special opcodes	197
24.6	Structures in code definitions	200
25.	Metacompiler	202
25.1	Concept of metacompilation	202
25.2	Vocabularies for metacompilation	204
25.3	Accessing memory in the target system	206
25.4	Branching constructs	207
25.5	Forward referencing	209
25.6	Compiling new words to target system	210
25.7	Transition compiler directives	211
25.8	Defining words in metacompiler	214
25.9	User variables	215
25.10	Vocabulary	216
25.11	Resolving forward references	217
25.12	Redefining host words	218
25.13	Running the metacompiler	219
	Index	222

FIGURES

1.1	The Forth family tree	2
1.2	The standard bearer	6
2.1	IBM-DOS files in F83 system	8
2.2	Forth words	10
2.3	Assembler and DOS words	11
2.4	Words in other vocabularies	12
2.5	VIEW and SEE	15
2.6	File and directory commands	17
2.7	Debugging LIST	19
3.1	Memory dump	26
4.1	The virtual Forth computer	37
4.2	Memory map of F83 system	39
6.1	Representation of strings	58
7.1	The file control block	61
7.2	Disk buffer management	65
8.1	Structure of a Forth definition	73
8.2	Vocabularies and dictionary structure	74
8.3	Four-way threading in a vocabulary	75
9.1	Input and output number conversion	82
10.1	Parsing with WORD	91
12.1	The interpreter and the compiler	101
13.1	Numeric data structures	106
13.2	The string literals	108
13.3	The control structures	110
15.1	The editing buffers	129
15.2	Screen editor display	135
16.1	The view field and the view files	142
20.1	Decoding different types of words	159
20.2	Decompile different words	164
21.1	Two printing formats	168
23.1	The round robin task scheduler	178
24.1	Register addressing mode constants	186
24.2	8086 instruction types	190
25.1	The Tao of meta-Forth	203
25.2	Supporting vocabularies for metacompilation	205

TABLES

2.1	Vocabularies in F83	9
3.1	Editor commands	22
3.2	Loading commands	24
3.3	8086 registers and Forth registers	28
3.4	Register addressing modes and mnemonics	28
3.5	8086 assembler commands, Forth style	29
3.6	Return commands	30
3.7	Machine code conditionals	30
4.1	8086 Register assignments for Forth	36
6.1	String commands	57
9.1	Data representation	80
23.1	User variables	176

(c) Copyright, 1991 by C. H. Ting

First Edition, November 1984

Second Edition, June 1985

Third Edition, June 1991

All rights reserved. This book, or any part thereof,
may not be reproduced in any form
without written permission from the author.

Printed in the United States of America

by

Offete Enterprises, Inc.

1306 South B Street
San Mateo, CA 94402

Tel: (415) 574-8250

INSIDE F83

Dr. C. H. Ting

Third Edition

Offete Enterprises, Inc.

1991