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 freware 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 oppotunity 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 3. Utilities in F83 system

Part 4. 8086 Specific utilities

FIGURES

TABLES

First Edition, November 1984

Second Edition, June 1985

Third Edition, June 1991

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