

Linear Interpolation

FIGUK magazine:

Choosing Forth

Competitive Programming With Forth

Book Review

Forth, the Early Years

Iteration with Many:

Source Code Index

Across the Big Teich

events

FIG UK – AGM 16

news

Forth News 2

reviews

**Book Review “The Practice of
Programming” 14**

Across the Big Teich 25

programming

Linear Interpolation 8

Iteration with Many: 19

people

Choosing Forth 5

Competitive Programming 7

Forth – The Early Years 18

Letters 32

projects

Source Code Index 23



Editorial

We approach our annual AGM with a full complement of officers and a reasonably healthy bank balance. FIG UK continues to provide a good service to its members. Please pass on your suggestions for improvements ahead of the meeting or, better still, come along in person.

I'm pleased to include another book review from Boris Fennema. Do any other members have a book they're keen to recommend?

FIG UK continues to innovate, launching a new service last month - the Forth Code Index - see details [here](#).

This issue contains two items which were refined through feedback from the newsgroup - you get the benefit of peer review.

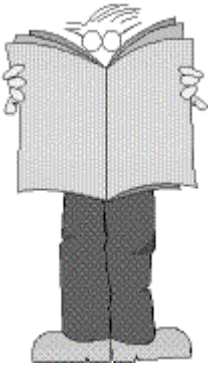
Congratulations to Bernd Paysan (of German FIG) for his Forth entry in the ICFP contest - see details [here](#).

Finally, we're delighted to announce another Corporate Member - Culver Consultancy Ltd.. Expect an interview in the next issue.

PS. Don't forget the monthly IRC session. Our next one is Saturday 5th October on the IRC server network called "IRCNet", channel #FIGUK from 9:00pm.

Until next time, keep on Forthing,

Chris Jakeman



Forth News

Forth Resources

Forth Code Index

In August FIG UK launched the Forth Code Index, the first index to published source code with a global coverage. This is on-line at our web site - see later in this issue for details.

FIG UK Enhances Web-Site

Jenny Brien has added a site map to the FIG UK web site at

<http://www.fig-uk.org/map.htm>

The site map includes a Google search of the site itself, especially useful for searching the archive of Forth News.

Getting Started

Dave Pochin has updated his long-established guide to getting started with Forth using Win32Forth. The update covers the latest version, Win32Forth v4.2067 at www.sunterr.demon.co.uk

TSCP Chess

Ian Osgood has announced 2 upgrades to his port of Tom Kerrigan's TSCP chess engine. These provide much higher performance, better documentation and some bug fixes. See

<http://ultratechnology.com/chess.html>

URL Monitor

Marcel Hendrix and Krishna Myneni have both published tools to monitor URLs such as web pages for changes.

Marcel's program makes use of the w3m browser, see

<http://home.iae.nl/users/mhx/scooter.ft>

and Krishna's program uses lynx and grep, see

<http://ccreweb.org/software/kforth/kforth4.html>

Electronic Filter

Marcel has also updated his FIR program for designing electronic filters at

<http://www.iaehv.nl/users/mhx/eprograms.html>

ANS Forth Published Papers

In the previous issue a link was provided to proposals for Internationalisation. Similar papers for cross-compiling can be found at:

<ftp://ftp.forth.com/pub/ANS%20Forth/>

Non-commercial Systems

.NET

Microsoft lists .NET Language Partners providing 16 languages that run on the .NET platform. The entry for the Forth language refers to Delta Forth shareware product from Valer Bocan.

"Delta Forth .NET integrates seamlessly into Microsoft's .NET platform and can interoperate with programs written in other .NET languages, such as C#, VB.NET, JScript, etc.. The Delta dialect is simple and easy to learn. It is excellent for academic environments and it's the perfect tool to write the scientific project you've always wanted."



Download Delta Forth v1.1 from <http://www.dataman.ro/dforth/index.html>

The site also records a dispute with Tucows, the software library site. Tucows alleged behaviour can only be described as bizarre.

PicForth for 16F87x

Sam Tardieu has published his free Unix (or Linux) hosted Forth compiler for the Microchip PIC 16F87x family. It is in no way finished yet (hence the low version number 0.1) but does a good job at generating efficient code. See

<http://www.rfc1149.net/devel/picforth>

Aztec Forth for Windows

Thomas Worthington has reposted his Aztec Forth for Win32 which is pretty small, hand-written in assembler and allows full access to the DLLs. See <http://www.tww.cx/azintro.php>

Ficl

John Sadler reports Ficl release 3.03 is now available for download at <http://ficl.sf.net>

Thanks to Daniel Sobral for this release, which includes a number of bug fixes.

Coming soon: Ficl 4.0 will feature a revised inner interpreter that's about twice as fast as the original.

kForth

A new version of kForth, version 1.0.12, is available for Linux and Windows at <http://ccreweb.org/software/kforth/kforth.html>

Commercial Systems

Global Positioning System

Triangle Digital Services have developed a Forth-based data logging system which collects location information from a GPS receiver. The photo shows the logger and the black hemisphere which is the GPS.



Application areas include agriculture, security, transport, boats etc.. Information that can be logged includes longitude and latitude, date and time, pressures, temperatures, rotation rates, flow rates, linear or rotational position, doors open/closed etc.

To recover data you can read the card in a PC, upload through a serial port or read the data directly into Excel spreadsheet.

Joining the F11-UK Mailing List:

Graeme Dunbar, our list moderator, has reported that the list has received a number of requests of uncertain origin to join it. Since spamming is a potential problem the banner on the List's home page has been altered to ask prospective members to identify themselves first. If you would like to join please apply to:

<http://groups.yahoo.com/group/fig-forth-uk/>

From the 'Net - Choosing Forth

This posting on comp.lang.forth encapsulates the Forth ideal so well, it was worth re-printing.

Subject: Re: Hang on, isn't Forth out of date now?

Date: Sat, 10 Aug 1996

Organization: Bell Global Solutions

Andy, thanks for the provocative question. It's helped me compose my thoughts for a flyer I'm writing.

No, Forth isn't out of date. Forth is still great because of the following benefits:

1. It is simple to build from the bottom up.
2. You can get an `_application_` to run in a miniscule amount of RAM.
3. You can try things out in real time as you build your system.
4. Compared to any other interpreted language, it is fast.

Allow me to elaborate:

1. Forth remains one of the few environments which is totally comprehensible by one person. This is a big plus when you're working in safety-critical systems, or whenever you need to verify program correctness.

2. Forth does indeed make "the best out of a slow microprocessor with little RAM." Such processors are more common than PCs -- they're called embedded systems. It will be a long time before your car's fuel-injection system has 16 MB and a 1.25 GB hard disk. (And most embedded processors are NOT supported by Borland C++.)

3. There is simply NO substitute for an interactive interpreter when debugging. Even an edit-compile-test cycle of 5 seconds feels clumsy, after you're used to testing any subroutine by typing its name. Can your debugger let you manually try different input parameters? (My Borland compiler can't.) And you should try a modern interactive

Forth to learn how easy it makes testing embedded hardware! I've yet to meet the in-circuit emulator that lets me exercise I/O as easily as a few simple lines of Forth code. (Or lets me test multiprocessors or distributed systems at all!)

4. Forth is still fast. Modern compilers produce code as good as any other language -- not all Forths use threaded code! (I could relate a horror story I heard about an engine control system written in C++.) Forth is certainly the fastest interpretive language around; and besides the debugging advantage, I've found interpretive Forth to be superior for incremental development.

However, I have other reasons for using Forth:

5. Forth is **extensible**. This means that if the language does not support some feature or capability you need, you can add it...not as a subroutine package, but as part of the language itself. Can you imagine writing object-oriented code, if every reference to an object had to be through a function call? That's how I feel about other languages' implementations of multitasking, multiprocessing, and networking. Only in Forth can these be truly transparent.

6. Forth lets me work at a high level of **abstraction**. Between language extension and "active" data structures, when I write a Forth application, I am really writing in the language of the application -- not the language of the compiler. This makes the program easier for a newcomer to read, and easier to maintain.

Like most programmers, my choice of language is based on personal preference. I find that I think more clearly in Forth and, from past experience, I estimate I'm 5 to 10 times more productive in Forth than in C. Others may not share this preference or facility. Forth may not be your preference, but it's certainly "relevant" -- now more than ever.

--

Brad Rodriguez bj@forth.org Computers on the Small Scale
This brain for rent! See <http://www.forth.org/fig/homes/brodriguez.html>
Contributing Editor, The Computer Journal.... <http://www.psyber.com/~tcj>
Director, Forth Interest Group..... <http://www.forth.org/fig.html>

Competitive Programming With Forth

Every year International Conference on Functional Programming holds a programming contest and publishes the results at the conference afterwards.

Or as C/Net put it on <http://news.com.com/2100-1001-956188.html> :

"While the rest of the nation lounges on the beach and bids farewell to summer, some computer programmers will spend the Labor Day weekend creating software robots that will deliver virtual packages while shoving each other into lethal ponds.

The occasion for this creative exercise is the Fifth ICFP Programming Contest, a 72-hour battle announced Friday and ending Monday at noon. Winning robots will square off against each other at the International Conference on Functional Programming held this year in Pittsburgh Oct. 4-6.

Entrants, vying for a \$1,000 cash top prize, may use any programming language they wish, but contest organizers are confident that the prevailing robot will be created with a functional programming language."

In 2001, there were 263 entries. The entries for this year have just been completed and will compete against each other at the conference. The table shows fewer entries from functional languages than from non-functional ones. However in past competitions, the functional language teams have performed well which should please the ICFP. As the table also shows, this year, and for the first time, there is an entry for Forth.

We have Bernd Paysan to thank for taking up the challenge on behalf of Forth - read his report at

The competition involves competing robots to operate within a grid defined by a simple character map. For example, here is a (very small) example board, where "#" represents an impassable wall and "~" indicates lethal water. The robots take it in turn to issue a simple text command (one of three possible ones). Speed of execution therefore is irrelevant and only quality of decision counts. The robots communicate with the game server using TCP/IP sockets. For details, see

<http://icfpcontest.cse.ogi.edu/task.html>

As Bernd explains, he wasn't able to spend the full 3 days on the contest, but he got his robot working so he submitted it anyway. Now he's set an example to Forth users everywhere, it would be great to see several Forth entries next year.

Language	Entries
Java	28
C++	23
C	21
Python	20
Ocaml	19
Perl	13
Haskell	10
Scheme	6
Lisp	4
Ruby	3
Erlang	2
Mercury	2
Delphi	2
Dylan	2
Ada	1
Forth	1
Icon	1
Prolog	1
Rice	1
SML	1
Smalltalk	1

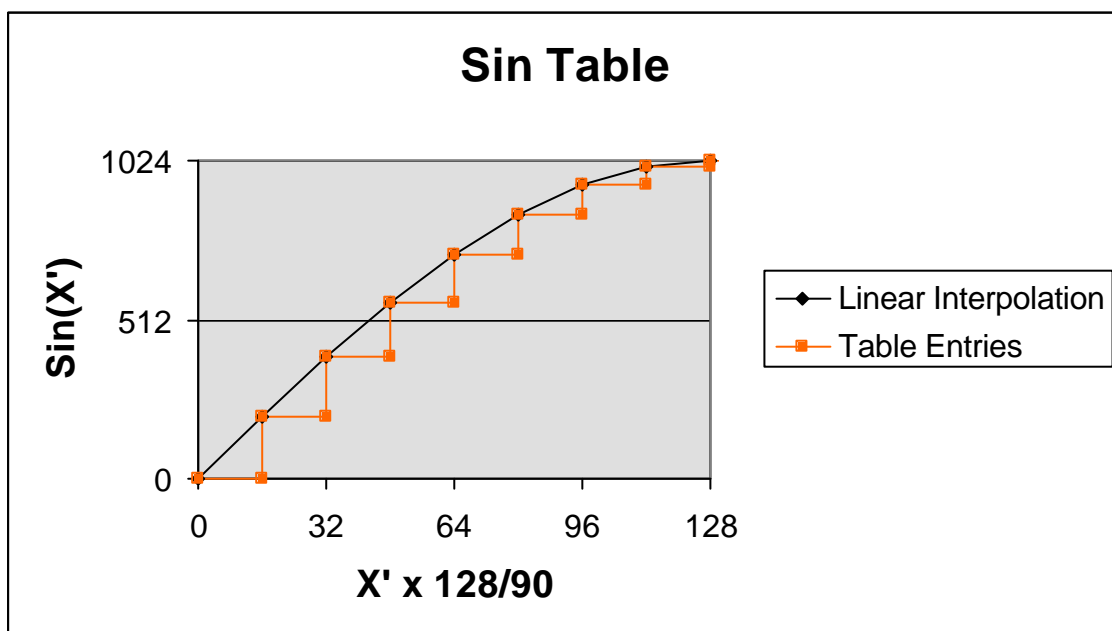
7	5
..@.....	
.....	
##.~~~~	
...~~~~	
.....	

Linear Interpolation

Chris Jakeman

The new Source Code Index (mentioned elsewhere in this issue) lists several ways to extract data from a table of data points and estimate the values in between the entries. This is called interpolation. With help from regulars on the newsgroup, Chris develops a fast solution which has not been previously documented.

The new Source Code Index lists several ways to extract data from a table of data points and estimate the values in between the entries. In this example, the data points provide the sine of an angle, with 8 data points are used to span a full right-angle. (Linear Interpolation is fast and suitable for any function, but do check that your Forth doesn't provide the function (eg FSIN) already.)



```
create SinTable \ Sample table for 2^10 x Sin(X*8/90)
\ Y      X      X'
  0 ,    \ 0.00  0
 128 ,   \ 11.25 1
 256 ,   \ 22.50 2
 569 ,   \ 33.75 3
 724 ,   \ 45.00 4
 851 ,   \ 56.25 5
 946 ,   \ 67.50 6
1004 ,   \ 78.75 7
1024 ,   \ 90.00 8
```

This first example shows how easy it is to build a data table and look-up a value.

```
: Sin ( Angle*8/90 - Sin{A}*1024 )
  cells SinTable + @
;
```

4 Sin . . (should = 724)

Once we add interpolation for values between the data points, the code gets more interesting. Brad Eckert has published a cubic interpolation (<http://www.tinyboot.com/cubic.txt>) which fits a curve between 4 points to extract the maximum accuracy out of the minimum number of data points.

At the other end of the spectrum, linear interpolation merely fits a straight line between 2 points (as in the chart) but this approach is much faster. Brad Eckert has also offered a linear solution which is complex but effective (<http://www.tinyboot.com/linear.txt>). It uses arithmetic on double numbers treating the top cell as an integer with the bottom cell as a fraction. I offered a shorter alternative on `comp.lang.forth` which uses two divisions; `/MDD` to choose the table entry and `*/` to calculate the result.

I am grateful to Marcel Hendrix for showing me how to devise a simpler, shorter solution which is more than 10 times faster than either of these. The solution offered here was refined on `comp.lang.forth` and owes its speed to the use of **RSHIFT** in place of division. The solution is presented twice, with the second version being a development of the first.

For the scheme to work, the X interval between data points must be a power of 2. In this example, it is 2^4 or 16. We do as much work as possible at compile time. Each entry in the data table contains 2 values, the Y value corresponding to X and also the $Y' - Y$ value, dY, which is the increase in Y to reach the next data point. We can fetch both values at once using `2@`.

```
: Entry, ( Y Y' -- Y' )
  swap 2dup - ( -- Y' Y Y'-Y=dY )
  ,          \ Compile dY
  ,          \ Compile Y
;

create SinTable \ Sample table for 2^10 x Sin(X*128/90)
\ Y      X      X'
  0      \ 0.00  0
  200 Entry, \ 11.25 16
  392 Entry, \ 22.50 32
  569 Entry, \ 33.75 48
  724 Entry, \ 45.00 64
  851 Entry, \ 56.25 80
  946 Entry, \ 67.50 96
  1004 Entry, \ 78.75 112
  1024 Entry, \ 90.00 128
drop
```

We first create **Entry**, as a convenient word to add entries to the table.

We will split the X value into two parts, the higher part matching one of the entries in the table and the lower part indicating the amount to be interpolated.

In this example, the interval between each X entry is 16 and X = 39. This is split into higher=32 and lower=7. **4 RSHIFT** is used as a fast divide by 16, giving an index down the table of 32/16=2.

We extract the two values corresponding to X=32, which are Y=392 and dY=177.

Finally, we divide dY by 16 to give the interpolation and add that to Y for the result.

```

4 CONSTANT Bits/Step    \ 2^4 = 16
15 CONSTANT BitMask     \ Ie. Binary 00001111

: Interpolate ( X*2^7/90 Table -- result*2^10 )
  swap dup BitMask and >r      \ Extract lowest 4 bits
  Bits/Step rshift            \ Fast divide by 16 gets index
  2 cells * +                 \ Index down the table
  2@                           \ Get pair of values for entry
  r> *                         \ Get interpolation
  Bits/Step rshift            \ Scale it
  +                             \ Add dY to Y
;

: Sin ( Angle*128/90 - Sin{A}*1024 )
  SinTable Interpolate
;

39 Sin . . ( should = 469 )

```

"simpler, shorter and 10 times faster"

A more refined version follows. By storing extra data at the start of the table, we can make it general, working with steps that are 2^N , not just 2^4 . It also includes an optional range check.

We improve accuracy by rounding the interpolation quantity before the final division, rather than truncating.

We also improve performance slightly by calculating **2 CELLS** at compile-time.

The earlier version didn't work for the final value X=128, as this wasn't actually included in the table data. To include this last value, repeat the last entry as shown below.

```

: Entry, ( Y Y' -- Y' )
  swap 2dup - ( -- Y' Y Y'-Y=dY )
  , \ Compile dY
  , \ Compile Y
;
create SinTable \ Sample table for 2^10 x Sin(X*128/90)
\ Preamble
  BitMask , Bits/Step , \ used by Interpolate
  0 , 129 , \ Min and max+1 limits to X
\ Data Points
\ Y X X'
  0 \ 0.00 0
  200 Entry, \ 11.25 16
  392 Entry, \ 22.50 32
  569 Entry, \ 33.75 48
  724 Entry, \ 45.00 64
  851 Entry, \ 56.25 80
  946 Entry, \ 67.50 96
  1004 Entry, \ 78.75 112
  1024 Entry, \ 90.00 128
  1024 Entry, \ Last entry repeated
drop

: Interpolate ( X*128/90 Table -- result*2^10 )
  swap \ -- Table X
  over 2@ \ Get Bits/Step and BitMask from table
  2 pick and \ Extract lower bits
  over >r >r \ Stash them and Bits/Step
  rshift \ Fast divide of higher bits gets index
  [ 2 cells ] literal * \ Index down the table
  [ 4 cells ] literal + \ Skip past the preamble
  + 2@ \ Get pair of values for entry
  r> * \ Calc interpolation
  r> 1- rshift 1+ 1 rshift \ Add 1 during division to round result
  + \ Add dY to Y
;

: SafeInterpolate ( X &Table -- Result )
  2dup 2 cells + 2@ within abort" Outside range of look-up table"
  Interpolate
;

: Sin ( Angle*8/90 - Sin{A}*1024 )
  SinTable SafeInterpolate
;

```

39 Sin . . (should = 469)

For maximum accuracy, aim for dY values that never exceed the dX value (although they do for most of the range of this example). Otherwise the interpolation will never return certain values of Y.

Also, if X is the result of other calculations, ensure these provide rounding to the nearest integer so that X=127.51 becomes 128 not 127.

Here is a small puzzle from Michael Gassanenko. A word with this behaviour is listed in the AS Forth core word-set. What is its name?

```
: X?  
  2DUP > TUCK INVERT AND >R AND R> OR ;
```

F11-UK

provides everything needed in a professional-quality low-cost Forth controller board.

Use it in industrial or hobby projects to control a wide range of devices using the well-known multi-tasking Pygmy Forth.

Designed for hosting from a Windows or DOS PC, you can test your application as it runs on the F11-UK board itself. The board was developed by FIG UK members to provide an easy way to explore the world of controlled devices – a niche where Forth excels.

The kit includes both hardware and software and is supported and sold to members at a nominal profit through a private company.

Software

PC-based PygmyHC11 Forth compiler running under DOS produces code for Motorola HC11 micro-controller.

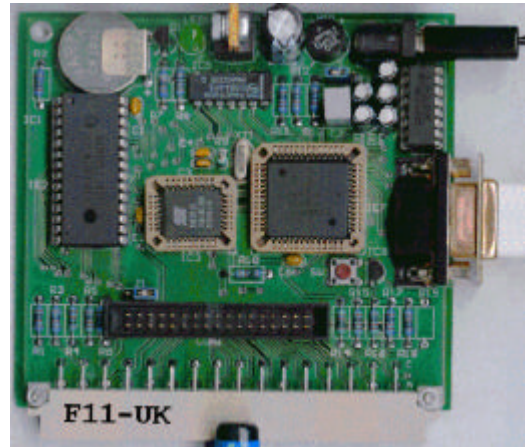
Code is downloaded via standard serial link from the PC to the FLASH memory (or RAM) on the F11-UK single board computer (SBC).

No dongle or programming adaptor of any kind is required.

Forth running on the SBC is interactive which makes debugging and testing much easier.

Multitasking and Assembly included.

The serial link can be disconnected to enable the SBC to function as a stand-alone unit.



All source code provided - 78 pages or so (unlike many commercial systems).

Around 30 pages of additional documentation is supplied including a full glossary of the 300 or so Forth words in the system.

Email mailing list for discussion and limited support.

Hardware:

Processor:

Motorola HC11 version E1 - 8 MHz (2 MHz E-Clock).

Memory:

32k x 8 FLASH

32k x 8 battery backed SRAM

512 x 8 EEPROM onboard HC11.

I/O:

20 lines plus 2 interrupts (IRQ & XIRQ).

Analogue in:

up to 8 lines using onboard 8-bit A/D.

Serial:

1. RS232, UART onboard HC11

2. Motorola SPI bus onboard HC11.

Expansion:

Via HC11 SPI serial bus using

2 or more of 20 available lines.

Timer system:

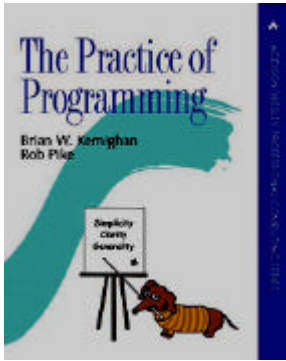
Inputs: 3 x 16-bit capture channels

Outputs: 4 x 16-bit compare channels.

PCB size: 103 x 100 mm.

Price to FIG UK members: £47.0 plus postage and packing (£2 UK, £4 overseas) plus \$25.0 (US Dollars) for registration of 80x86 Pygmy Forth with the author Frank Sergeant.

Delivery: ex-stock.
More information: jeremy.fowell@btinternet.com and 0121 440 1809



fennema@gofree.indigo.ie

Book Review "The Practice of Programming"

Boris Fennema

Here is another welcome book review from Boris.

Introduction

This book covers the concepts of good programming practices – " topics like testing, debugging, portability, performance, design alternatives, and style -- the *practice* of programming -- are not usually the focus of computer science or programming courses. Most programmers learn them haphazardly as their experience grows, and a few never learn them at all. "

To do so, the authors illustrate their examples in a number of languages C, C++ and Java and scripting languages (AWK mainly). The authors speak with some authority; Brian Kernighan is the co-author of the C programming language and represents the "K" in the AWK language. They have collaborated on joint books before.

The Practice of Programming
Brian W. Kernighan and Rob Pike
Addison-Wesley, 1999
ISBN 0-201-61586-X
pp256, £11.49 from www.amazon.co.uk

Their main thrust is that a well-written program should observe four guidelines:

- simplicity
- clarity
- generality
- automation

The authors maintain that a program written with these in mind will be easier to create, understand, maintain and revise and will be a better program all round.

The following aspects of programming practice are covered in 9 chapters: style, algorithms and data structures, design and implementation, interface design, debugging, testing, performance, portability, and notation. The book's appendix includes a collection of rules based on these 4 principles. The intended audience are people learning to program and people involved in programming directly as programmers or indirectly as managers or support staff.

Review

The first two chapters set the scene. The first chapter highlights the importance of good style and naming when writing source code. The next chapter reviews some fundamental data structures and some sorting and searching algorithms. It also introduces the $O(n)$ notation which will be used when discussing performance.

Chapter 3 discusses a program to generate text from an input text, but the output is altered via a Markov Chain. The algorithm ensures that the output text resembles the input text, but with some randomness thrown in. After the algorithm is discussed, it is implemented in C, C++, Java, Perl and AWK and the differences in implementation and performance discussed. The reader is invited to implement the program in a language of his/her choice and see how it compares (Forth anybody?).

The fourth chapter provides an good insight into how hard it is to build a production-strength utility library. It discusses library issues such as public access, information hiding, resource and memory management and error handling. The discussion is kept focussed by using a utility library to parse Comma Separated Values (CSV) – this makes the chapter very engaging to follow without being too long or too abstract. It finishes with some guidelines on writing modular code.

The fifth and sixth chapter give useful guidelines on debugging and testing techniques which are very useful for novice programmers. To have all the techniques in one book makes it a useful refresher course for the hardened programmers among us.

The performance and portability chapters are also useful, albeit specific to C/C++.

The last chapter focuses on notation. This is an area which would be recognised by Forth'ers as very powerful as, where a language notation can be matched to the problem, it can be expressed more simply and elegantly. It describes regular expressions, marshalling of data (describing a layout of a structure with a simple type grammar).

It also shows that if the problem can be broken into stages and a specific tool applied to each of the sub-problems, it can be solved quickly by using the leverage of each of the tools. If needed, the problem could be re-expressed in a little language and a specific compiler invented for this new language.

Summary

This is a useful book – it covers the 4 principles of clarity, generality, simplicity and automation well. The main focus in its examples is on C/C++ but they are still appropriate to other languages.

It shows the strengths of having a few appropriate languages in your programmer's tool box. Applying the correct language in the proper style to a problem can make big differences – (since Forth is flexible in syntax, it can play a role where languages with more strict grammars fail).

I did decide after reading it to invest some time in learning AWK and TCL (for some reason I cannot get my head around Perl) which in itself was well worth the effort.

FIG UK – AGM

Our Annual General Meeting will be held on Saturday 19th October at Doug Neal's home, 58 Woodland Way, Morden from 2:00pm.

All members are cordially invited to do attend. If you cannot come, but wish to comment on the way FIG UK is going or the direction you would like it to take, write or e-mail Jeremy or myself before the meeting.

Anyone who lives in the London area can get to Doug's house easily by Underground as he is just ten minutes walk from the southern terminus of the Northern Line. You can get directions from <http://www.multimap.com> for his postcode SM4 4DS or just phone him on 020 8542 2747.

Some of the topics likely to be discussed are:

- Joint projects such as Quikwriter Project
- Ideas for the Web Site
- Finances – see annual accounts on next page
- EuroFORTH 2003
- Promoting Forth and FIG in the UK

The previous issue of Forthwrite included a review of "Writing Your Own Programming Language" by Norman Smith. Boris Fennema reports that this is available from <http://www.amazon.co.uk> for £13 (new) or £6 (used).

Forth Interest Group UK: Revenue Account for year ending 31 March 2002

The information from these accounts will be discussed at the forthcoming AGM to which all members are invited and reported in the next issue.

Income and Expenditure

Y/E 31/3/01		Y/E 31/3/02	
1,293	Subscriptions	1,111	
	Advertisements	144	
8	Interest (net of tax)	4	
<hr/>		<hr/>	<hr/>
	1,301		1,259
593	Printing Forthwrite	649	
215	Postage	245	
26	Other Printing		
50	Library Expenses	55	
	Website Expenses	80	
		<hr/>	<hr/>
	884		1,029
	<hr/>		<hr/>
	417 Net surplus for the year		230

Balance Sheet as at 31 March 2002

Y/E 31/3/01		Y/E 31/3/02	
527	Accumulated Fund b/f	944	
417	Surplus for the Year	230	
<hr/>		<hr/>	
	944		1,174
Represented by:			
1,580	Cash at Bank	1,792	
	Unexpired Expenditure	11	
		<hr/>	<hr/>
			1,803
586	Unexpired Subscriptions	618	
50	Sundry Creditors	11	
<hr/>		<hr/>	<hr/>
	636		629
	<hr/>		<hr/>
	944 Net assets		1,174

Note: The subscription is treated as representing six issues of Forthwrite (five were published in the year). No value has been placed on unsold back numbers, the library or other stocks.

29th August 2002
Marlowe House, Hale Road,
Wendover, Buckinghamshire

NEVILLE A. JOSEPH
Chartered Accountant

Forth – The Early Years

"The Evolution of Forth", a 46-page history of the first 20 years of Forth programming by Rather, Colburn and Moore was presented at the ACM SIGPLAN History of Programming Languages Conference 1993, printed in Forthwrite (April 1995) and can now be found at <http://www.forth.com/Content/History/History1.htm>

Chuck Moore uses a rather different style¹ in his personal history of the development of Forth up to the formation of Forth Inc. in 1973.

In particular, he traces the development of each group of words in the Forth dictionary, showing how the applications he was writing each required an extension to the core until he felt Forth had grown into a complete entity.

Chuck also mentions the issue of software patents and the strong possibility that Forth would not have been released into the public domain.

This history can be found on-line at <http://www.colorforth.com/HOPL.html>

<u>Functionality Time Line</u>		
SAO	1958	Interpreter
SLAC	1961	Data stack
RSI	1966	Keyboard input Display output, OK Editor
Mhasco	1968	Compiler Return stack Dictionary Virtual memory (disk) Multiprogrammer
NRAO	1971	Threaded code Fixed-point arithmetic

¹ Chuck writes: "This paper was written for the History of Programming Languages Conference. It was summarily rejected, apparently because of its style. Much of the content was included in the accepted paper – The Evolution of Forth."

Iteration with Many:

Leo Wong and Chris Jakeman

A thread on comp.lang.forth² asked how best to define a sequence of integer constants – an enumeration. Responses ranged from the very simple to the very general. Leo proposed **Many:** as a middle-of-the-road solution and offered several examples showing that it has uses beyond enumeration.

Here are some postings from the thread:

John Drake (drake@cis.uab.edu)

Recently I was working on a project and needed an enumerated data type. I found two approaches to this at the Taygeta Forth repository. Both worked by defining a way to create successive constants. The first created a defining word that kept track of the next constant value.

```
: ENUM ( -- )
  CREATE 0 ,    DOES> ( -- n )  DUP @ CONSTANT 1 SWAP +!
;
```

An example of use is:

```
ENUM COLOR
COLOR RED
COLOR BLUE
COLOR GREEN
```

ENUM defines a new enumerate called COLOR. COLOR can then be used to define unique CONSTANTs for each color. In the example, RED will return a value of zero, BLUE will return 1, and GREEN will return 2.

That's ok, but I'm lazy. The thought of having to type:

```
ENUM MONTH
MONTH JAN  MONTH FEB  MONTH APR ...
```

etc. was a bit much for me. Then I found another enum example by Everett Carter at <ftp://ftp.taygeta.com/pub/Forth/Tools/enums.fth>

This one worked as follows:

```
12 enums: jan feb mar apr may jun jul aug sep oct nov dec
```

² Thanks to the regular posters on the newsgroup for helping to refine Many:

That was ok. But what happens when the enums span more than one line? (The problem I was working on needed 68 enums). Also I didn't like the fact that I had to keep up with the number of enums that needed to be created.

[Elizabeth D Rather](mailto:erather@forth.com) (erather@forth.com) responded with an approach which is very simple but doesn't save John from typing **ENUM** exactly 68 times:

SwiftForth's version is even simpler:

```
: ENUM ( n -- n+1 )
  DUP CONSTANT 1+
;
```

So you start the sequence wherever you like:

```
1 ENUM JAN  ENUM FEB  ENUM MAR  ... DROP
```

and lines, etc., aren't a problem. No need to be complicated about this.

Wil Baden (wilbaden@netcom.com) has published a comprehensive solution he calls Iterative Interpretation, see Forthwrite June 1999.

```
1 (: ENUM || JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC :)
DROP
```

which avoids repeating **ENUM**. The key point here is that Wil separates the repetition provided by (: ... | ... | ... :) from the phrase **ENUM** that is being applied. This makes his solution useful in a much wider range of situations.

Leo Wong suggested a less capable but simpler word **MANY** which also separates repetition from application. John would use it as

```
: enums ( n -- n+1 ) DUP CONSTANT 1+ ;
1 many enums jan feb mar apr may jun july aug sep oct nov dec
drop
```

This early version reads only to the end of a line and a second word **MORE** was needed to read each additional line. With feedback from the newsgroup, Leo has now refined **MANY:** to read multiple lines until it meets ";" – its terminator. (**MANY:** has acquired a trailing ":" to avoid confusion with **MANY**, a "Bon Mots" from Forthwrite March 1998).

Many: will also skip over comments indicated by (..) and \

Many: might be used as follows.

- to suit John Drake:

```
: enums ( n -- n+1 ) DUP CONSTANT 1+ ;  
  
1 many: enums jan feb mar apr may jun  
jul aug sep oct nov dec ; drop
```

- to save on typing

```
: constants ( -- ) <next> EVALUATE CONSTANT ;  
  
many: constants  
3 three  
2 two  
1 one  
0 zero  
;
```

- to compile text into the Data Space

```
: string, ( a u -- ) DUP C, 0 ?DO COUNT C, LOOP DROP ;  
: wrds ( -- ) <next> string, ;  
CREATE speech  
  
many wrds Four score and twenty years  
ago our fathers brought Forth  
on this continent.... ; 0 C,
```

which can then be printed using

```
: say ( a -- )  
BEGIN COUNT DUP WHILE 2DUP TYPE SPACE CHARS + REPEAT 2DROP ;  
  
speech say
```

- to create fields for structured data which remember their offset and size

```
: field ( offset -- offset+length )  
<next> EVALUATE 2DUP 2CONSTANT CHARS + ;  
  
0 many fields  
9 ssn  
20 firstname  
15 lastname  
35 street  
15 city  
2 ny ( state )
```



```
9 zip \ etc.  
; .
```

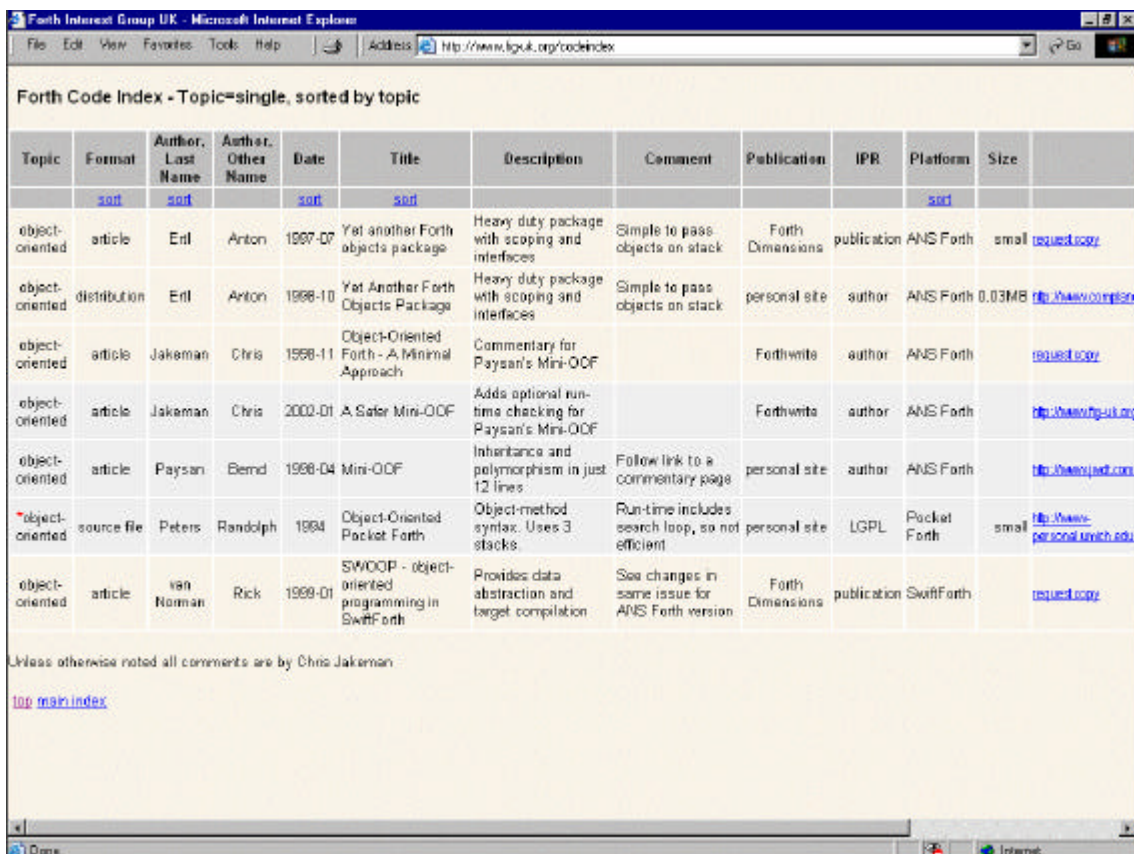
Many: is defined below. **preparse** looks ahead to the next word but rewinds the input stream ready to re-read the word it found.

```
: <next> ( -- a u ) BL WORD COUNT ;  
: preparse ( -- a u ) >IN @ >R <next> R> >IN ! ;  
: many: ( ... -- ... ) \ Usage: many: <word> ... ;  
' >R  
BEGIN  
  preparse 2DUP S" ;" COMPARE  
WHILE  
  2DUP S" (" COMPARE 0= IF 2DROP POSTPONE ( ELSE  
  2DUP S" \" COMPARE 0= IF 2DROP POSTPONE \ ELSE  
    NIP IF R@ EXECUTE ELSE  
    REFILL 0= ABORT" ; missing after Many:"  
  THEN THEN THEN  
REPEAT 2DROP  
BL WORD DROP \ Skip the ";"  
R> DROP ;
```

Source Code Index

Following a substantial thread³ on the comp.lang.forth newsgroup about promoting a "culture of sharing", FIG UK has published an on-line Source Code Index. This is hosted on our web-site at <http://www.fig-uk.org/codeindex/> and provides references to as many items of published source code as possible. This is the first ever index of source code to have a global remit and we hope it will become a valuable service from FIG UK to the Forth community.

The Index begins with a summary page dividing the entries into topics. You can examine the entire index or a single topic sorted by title, author, date, platform or format. Here is a sample page for the topic "object-oriented".



The screenshot shows a web browser window titled "Forth Interest Group UK - Microsoft Internet Explorer" with the address bar displaying "http://www.fig-uk.org/codeindex". The page content is titled "Forth Code Index - Topic=single, sorted by topic". It features a table with 12 columns: Topic, Format, Author, Last Name, Author, Other Name, Date, Title, Description, Comment, Publication, IPR, Platform, and Size. The table lists several entries related to object-oriented programming in Forth, including "Yet another Forth objects package", "Object-Oriented Forth - A Minimal Approach", and "SWOOP - object-oriented programming in SwiftForth". Each entry includes a link to a request copy or the source code.

Topic	Format	Author, Last Name	Author, Other Name	Date	Title	Description	Comment	Publication	IPR	Platform	Size	
object-oriented	article	Enl	Anton	1997-07	Yet another Forth objects package	Heavy duty package with scoping and interfaces	Simple to pass objects on stack	Forth Dimensions	publication	ANS Forth	small	request copy
object-oriented	distribution	Enl	Anton	1998-10	Yet Another Forth Objects Package	Heavy duty package with scoping and interfaces	Simple to pass objects on stack	personal site	author	ANS Forth	0.03MB	http://www.complex...
object-oriented	article	Jakeman	Chris	1998-11	Object-Oriented Forth - A Minimal Approach	Commentary for Paysan's Mini-OCF		Forthwrite	author	ANS Forth		request copy
object-oriented	article	Jakeman	Chris	2000-01	A Saker Mini-OCF	Adds optional run-time checking for Paysan's Mini-OCF		Forthwrite	author	ANS Forth		http://www.fig-uk.org
object-oriented	article	Paysan	Bernd	1998-04	Mini-OCF	Inheritance and polymorphism in just 12 lines	Follow link to a commentary page	personal site	author	ANS Forth		http://www.pact.com
*object-oriented	source file	Peters	Randolph	1984	Object-Oriented Packet Forth	Object-method syntax. Uses 3 stacks.	Run-time includes search loop, so not efficient	personal site	LGPL	Pocket Forth	small	http://www-personal.slvch.edu
object-oriented	article	van Norman	Rick	1999-01	SWOOP - object-oriented programming in SwiftForth	Provides data abstraction and target compilation	See changes in same issue for ANS Forth version	Forth Dimensions	publication	SwiftForth		request copy

Unless otherwise noted all comments are by Chris Jakeman.

[top main index](#)

The right-hand column provides a URL link to visit a web-page, download a file or send an email request for a photocopy from a publication.

³61 postings kicked off by John Passaniti - see http://groups.google.com/groups?as_umsgid=uj6bcvafpj89d@corp.supernews.com

Unusually, the Index includes fields for Description and Comment, indicating the scope of each entry and giving a hint to its likely value. The downside is that every item has to be read and appraised; also the author may dispute the compiler's opinion. Let's hope the benefits outweigh the brickbats.

The index was seeded with articles from Forthwrite, Forth Dimensions, tutorials and source files from member Krishna Myneni's extensive collection. There are currently 243 entries in 34 topics and items are being added from <ftp.forth.org> and <forth.sourceforge.net>.

When you visit the Index page, please mention any material that is still missing to the organisers. Also, note the topics where there are few entries or none at all. Could you make a contribution?

Building the Index was a challenge. In theory, all the data could be stored in an XML file and a subset of the data retrieved, sorted and presented using an XSLT script.

Learning all the procedures to implement this for the FIG UK web site server seemed too much of a learning curve, so the current implementation is simply a large set of similar HTML pages with the same data sorted in different ways.

But it's fast and suitable for any browser. Changes are made in a single place and the multiple pages are then updated from a script.

Across the Big Teich

Henry Vinerts

This material was prepared for Vierte Dimension by Henry Vinerts, and printed by kind permission of Forth Gesellschaft (German FIG)

FIG Silicon Valley Chapter Meeting - June 2002

Greetings from Northern California!

On June 22nd the SVFIG met and celebrated the arrival of another sunny summer, meteorologically speaking, at least. The next day's headline in the San Jose Mercury News, the leading local newspaper, read: "High-tech slowdown may be worsening", and "Silicon Valley's high-tech hangover isn't over. In fact, it may be getting worse." According to the paper, unemployment in Santa Clara County rose from 1.7% in year 2000, to 7% in 2001.

I suspect that the typical SVFIG member falls outside these statistics. In the past few years the attendance at our meetings has not fluctuated markedly, nor have there been an unusual number of inquiries for employment. It would be interesting to collect some statistics from the "Forthers" to see how and why they differ from the local mainstream and whether there are similarities among them on a global scale.

Dr. Ting certainly has been busy, mainly with Forth in Taiwan. His report filled the two hours in the morning without any breaks. First, about his P8 and P16 prototype chips, which have been made in Taiwan, and how they will work with eForth. Then, how the Nintendo GameBoy Advance with its ARM7 microprocessor (see <http://www.gameshark.com>) serves as a very nice platform for Forth and provides portable technology for Chinese character generation.

Although Win32Forth would work on this system, Ting says that he found Win32Forth too complicated and never could write a good manual for it. Therefore he developed F#, a 32-bit, protected mode, subroutine-threaded eForth implementation with a Windows interface.

With F# he is helping to "fine-tune," or aesthetically adjust the 70,000 Chinese characters that are generated by a Taiwanese program running on Windows XP (see also report of next meeting).

After the lunch hour (and that was the only break until quitting time at 4 P.M.), Scott, a first-time visitor, related his 20 years of work experiences, starting with Forth (MVP, F83) in machinery controls and ending-but not yet being finished-with Forth (written in TCL!) in his attempts to generate web-page-presentable automated error-code finders for troubleshooting ladder-logic systems, such as are used in PLCs (programmable logic controllers).

It was good to see the ex-president of FIG, Skip Carter. Could it be that the slumping economy has granted him some time for Forth again? Skip concluded our meeting with a demonstration of a brand new PDA (personal digital assistant) from Sharp in Japan, the "Sharp Personal Mobile Tool" SL-5000, which costs less than \$600 in the USA. Since it comes ready to run Linux and the GCC cross-compiler from Sharp is free, Skip has wasted no time to port Anton Ertl's Gforth to it.

Using a wireless browser (from <http://www.caphnet.com>, I believe) that runs on PDAs and cell phones, Skip can use the SL-5000 with a wireless network card for Internet access from practically anywhere. For more details, go to Skip's web site,

<http://www.taygeta.com/forthcomp.html>

Incidentally, Skip said that in Japan cell phones can do PDA work. I wonder how long it will be before cell phones replace the backpacks that children carry to school with them? Or make it unnecessary to go to school at all? So far, however,



cell phones have failed to help fight the tremendous forest fires blazing in the U.S.A., but I would not be surprised that more money is being spent on them than on relieving national disasters.

Carpe diem!

Henry

FIG Silicon Valley Chapter Meeting - July 2002

Hello, everybody!

The world has changed, but SVFIG marches on as before. As a matter of fact, I could probably send you one of my old reports in lieu of this one and not raise any suspicions among the readers.

Dr. Ting did his usual stint of nearly two solid hours in front of the audience. Besides the indispensable coffee pot, he had also lugged in a complete Compaq desktop, the only computer he could find that was running Windows XP. This was needed for the demonstration of Chinese characters that can be generated with the help of F#, which is yet another Forth dialect--an implementation of eForth, developed for XP by a young computer scientist in Taiwan.

Windows XP itself comes with about sixty thousand Unicode Chinese characters, all of which can be constructed from perhaps 1,000 low-level components or strokes. F# generates the strokes and stores them in bit-map form in less than one megabyte of memory.

As such, ported to Nintendo's GameBoy Advance which comes with 8 MB of flash memory and a 256x256 pixel display, F# is hoped to be the best way to surmount the biggest barrier which Chinese youngsters have to learning programming - the need to know English.

Dr. Ting's demo included the GameBoy with various displays of the Chinese characters. His most recent work is still centered around enriching the single-line strokes to look more like brush strokes, i.e., changing the widths of the lines at each point. Thus a bonus lecture on Bezier curves, quadratic and cubic

equations, enough to make my head swim.

Next came a brief announcement by John Peters of his idea to collect onto one CD all Forth words that have ever been written and, with the help of a heap-search mechanism, use it for encyclopedic purposes. A number of people have already volunteered for this project and additional helpers are invited.

I don't believe that at any time of the day my count of the attendees exceeded 20, but after lunch the meeting seemed to go pretty enthusiastically.

The little time left after the afternoon break barely allowed Dennis Ruffer to finish his talk about his work at Apple Computers. According to him, the only Forth jobs that are offered on the Web are for F-code programming in the Open Firmware boot-up process, which by now is on every Macintosh computer. Since Gforth is used to compile the boot ROM, Dennis fielded a lot of questions about this Forth, but he could not tell us what the "G" stands for (GNU).

This brings me to my own idea for a project. I would like to see a list of all of the Forths that have been named over the years, with the dates of origin, names of authors, explanations of the names, country or language of origin, major applications, etc.. I suppose the database could include relatives like Postscript, FIFTH, etc, with proper explanations. How about some input on this for me? I would like to give John Peters an idea of how many waters to fish in for Forth words.

Henry

FIG Silicon Valley Chapter Meeting - Aug 2002

Hello,

Bob Nash was the only scheduled speaker for this month's meeting and had traveled about 100 miles from Sacramento (the State Capital) to tell us about his work with SwiftForth. The Sacramento

Municipal Utilities District (SMUD) has been using a number of Forth Inc., products in real-time applications since the early 1990s. The full-blown SwiftForth, running on Windows, costs about \$1500, but evaluation copies are freely available from Forth Inc., either from their web site or by mail. There is an SFTalk group on the web, hosted by Rick Van Norman, whom Bob calls the "support genius".

Alan Furman added to the lecture with an interesting demo of one of his applications of SwiftForth. After lunch there were a number of short informal announcements and presentations, a prolonged rest break, and last, but not least, a 10-person "round-table" discussion led by Dr. Ting on the subject of preferred methods of data storage and retrieval. Dr. Ting maintains that blocks and files, like editors, are not fundamental to Forth.

Wishing a good time to everyone who is taking a vacation, intentionally or otherwise, I remain yours truly,

Henry



Alan J M Wenham
01932 786440
101745.3615@compuserve.com

Vierte Dimension *Alan Wenham*

Dear Chris,

I have been contributing summaries of Vierte Dimension for several years now and it has been a great pleasure to do so. However, my personal and domestic circumstances have changed and I find that I am no longer able to continue with this task.

I hope that you will be able to find somebody to take my place, because I consider the German Forth scene to be very important.

Yours sincerely,

Alan Wenham

Alan has been keeping us in touch with Forth events and developments in Germany. Over the past 4 years, he has been a pleasure to work with.

In a feedback exercise recently, we found that these efforts were much appreciated by members, so I am delighted to be able to announce that Joe Andersen has offered to continue the work that Alan began. Joe lived for some years in Munich and we look forward to his first report in the next issue.



Deutsche Forth-Gesellschaft

Would you like to brush up on your German and at the same time get first-hand information about the activities of fellow Forth-ers in Germany?

Become a member of the German Forth Society for 80 DM (£28) per year (32 DM (£11) for students and retirees). Read about programs, projects, vendors and our annual conventions in the quarterly issues of *Vierte Dimension*.

For more information, please contact the German Forth Society at the e-mail address SECRETARY@ADMIN.FORTH-EV.DE

or visit <http://www.forth-ev.de/>

or write to

Forth-Gesellschaft e.V.

Postfach 161204

18025 Rostock

Germany

Tel.: 0381-4007872

Letters

The Magazine Team are always pleased to get feedback and encouragement. The first letter comes from an old member, well-known for his innovative implementation of Forth on Windows - Aztec.

**Thomas
Worthington**

Hi Chris,

Anyway, I'll drop you a line once Aztec is back up. [See Forth News, Ed] I started on a Linux version but it ground to a halt as I tried to decide WHICH Forth I want to use under Linux: ANS or Colo(u)r. I'm interested by Chuck's current system but I have grave doubts about applying it in an OS setting where the purity he has achieved has to be compromised to fit into the system.

On the other hand, I've never really been happy with ANS, in particular the difficulty of manipulating the return stack to produce code such as the old BNF parser.

I got as far as simple colon-style definitions (in 8K) and started dithering and I've decided to wait until I have a design ready instead of just typing whatever comes to mind!

Thomas

This letter comes from a new user of the F11-UK Forth controller kit.

**Phillip
Eaton**

Hello All,

I've just joined the group, so here's a quick introductory message.

In a previous, happier, life I spent several years working on commercial SCADA control and monitoring projects using Forth-based systems, both embedded (HD64180/Z80/8086) and on the PC, for a company called Lee-Dickens Limited. See http://www.lee-dickens.co.uk/systems/prod_rtu.htm for information.

My interest in Forth is extended by my other hobby, restoring old arcade games from the late 70's early 80's. Running on 6502, Z80 and 6809 hardware, these are the perfect hardware for implementing a Forth operating system for developing new programs (especially for testing the hardware) and generally having some fun with.

Due to other current commitments, I don't have much time for Forth activity, but when I do get a minute, I'm also developing an Atari Centipede game emulator (6502 emulation based) on the PC using MPE PowerForth.

See it here www.phillipeaton.com/personal/arcade.htm.

Thanks for your time,
Phillip Eaton

Our final letter comes from an academic based in Ireland who published research into Forth. last year. For a full list, see <http://www.cs.may.ie/~jpower/Research/Papers/>
James invites anyone with an interest in this field to contact him via james.power@may.ie

**James
Power**

> Hi James,
>
> I am the editor of Forthwrite magazine, the journal of FIG UK. I
> found your paper on the internet and also your list of publications
> and was wondering whether any work was continuing in this field.
>
> I recognise your references to Peter Knaggs and Bill Stoddart (a
> FIG UK member). Would you mind explaining why you chose Forth
> as the subject of your investigation? Was it connected with the
> simplicity of Forth or were there other reasons?

Hi Chris,

Thanks for your note and interest in our work.

Really we were working on Forth with one eye on the Java Virtual Machine (JVM) which incorporates stack-safety verification as part of its dynamic class loader.

David and I were interested in developing an approach that could provide an environment for verifying similar properties of Forth programs. The basic idea is to try and statically verify (where possible) that a program won't cause a stack error at runtime.

We haven't moved the work significantly forward since the paper you mention. I guess our next step would be to try out our model on some "real world" Forth programs and see if we can sensibly verify some properties. We'd certainly be interested in hearing from anyone with a related interest.

James Power



Chairman **Jeremy Fowell,** 11 Hitches Lane, EDGEBASTON B15 2LS
0121 440 1809 jeremy.fowell@btinternet.com

Secretary **Doug Neale,** 58 Woodland Way, MORDEN SM4 4DS
020 8542 2747 dneale@w58wmorden.demon.co.uk

Editor **Chris Jakeman,** 50 Grimshaw Road, PETERBOROUGH PE1 4ET
01733 352373 cjakeman@bigfoot.com

Treasurer **Neville Joseph,** Marlowe House, Hale Road, WENDOVER HP22 6NE
01296 62 3167 naj@najoseph.demon.co.uk

Webmaster **Jenny Brien,** Windy Hill, Drumkeen, BALLINAMALLARD,
Co. Fermanagh BT94 2HJ
02866 388 253 webmaster@figuk.plus.com

Librarian **Graeme Dunbar** Electrical Engineering, The Robert Gordon University,
Schoolhill, ABERDEEN AB10 1FR
01651 882207 g.r.a.dunbar@rgu.ac.uk

Membership enquiries, renewals and changes of address to Doug.
Technical enquiries and anything for publication to Chris.
Borrowing requests for books, magazines and proceedings to Graeme.

FIG UK Web Site

For indexes to Forthwrite, the FIG UK Library and much more, see <http://www.fig-uk.org>

FIG UK Membership

Payment entitles you to 6 issues of Forthwrite magazine and our membership services for that

period (about a year). Fees are:

National and international	£12
International served by airmail	£22
Corporate	£36 (3 copies of each issue)

Forthwrite Deliveries

Your membership number appears on your envelope label. Please quote it in correspondence to us. Look out for the message "SUBS NOW DUE" on your sixth and last issue and please complete the renewal form enclosed.

Overseas members can opt to pay the higher price for airmail delivery.

Copyright

Copyright of each individual article rests with its author. Publication implies permission for FIG UK to reproduce the material in a variety of forms and media including through the Internet.



FIG UK Services to Members

- Magazine** Forthwrite is our regular magazine, which has been in publication for over 100 issues. Most of the contributions come from our own members and Chris Jakeman, the Editor, is always ready to assist new authors wishing to share their experiences of the Forth world.
- Library** Our library provides a service unmatched by any other FIG chapter. Not only are all the major books available, but also conference proceedings, back-issues of Forthwrite and also of the magazine of International FIG, Forth Dimensions. The price of a loan is simply the cost of postage out and back.
- Web Site** Jenny Brien maintains our web site at <http://www.fig-uk.org>. She publishes details of FIG UK projects, a regularly-updated Forth News report, indexes to the Forthwrite magazine and the library as well as specialist contributions such as "Build Your Own Forth" and links to other sites. Don't forget to check out the "FIG UK Hall of Fame".
- IRC** Software for accessing Internet Relay Chat is free and easy to use. FIG UK members (and a few others too) get together on the #FIG UK channel every month. Check Forthwrite for details.
- Members** The members are our greatest asset. If you have a problem, don't struggle in silence - someone will always be able to help. Do consider joining one of our joint projects. Undertaken by informal groups of members, these are very successful and an excellent way to gain both experience and good friends.
- Beyond the UK** FIG UK has links with International FIG, the German Forth-Gesellschaft and the Dutch Forth Users Group. Some of our members have multiple memberships and we report progress and special events. FIG UK has attracted a core of overseas members; please ask if you want an accelerated postal delivery for your Forthwrite.