# Forthwrite

December 2003
Issue 123
ISSN 0265-5195

news events people reviews projects programming

## A Virtual Nondeterministic Machine in Forth

FIGUK magazine:

AGM Report

EuroForth 2003 – The Report

Across the Big Teich

Vierte Dimension 2/2003

(inside front cover – blank)

**Forthwrite**
**December**
**2003**
**Issue 123**
**ISSN 0265-5195**

# Editorial

Well my second issue came around sooner than expected, so my apologies for it being so late...

Hopefully this issue will drop on to your doormat in time for the Committee to wish you a Happy New Year!

In this issue James Boyd writes about virtual nondeterministic machines in Forth; a fascinating subject and an interesting approach to solving unwieldy search space problems. Howerd Oakford of Inventio Software Ltd gives a lively account of this year's EuroForth conference.

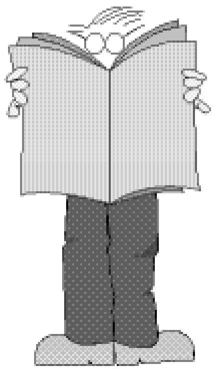
One of the topics discussed at the AGM was the future of Forthwrite. For it to exist in its present form we really need contributions from members. Requests for help and topics for discussion, observations and comment are just as welcome as articles. Looking back over past volumes, Forthwrite used to contain a sizable number of letters on a wide range of topics, so if you don't feel up to writing a full article then a letter is the painless alternative.

In an attempt to get back on course I have already started work on the first issue of 2004. It might help potential authors if the next couple of copy deadlines were published, so the next two are shown on the inside back cover. If you have any material for publication in the next issue please send it to me by Wednesday 21st January.

PS. Don't forget the monthly IRC session. Our next one is Saturday 3rd January 2004 on the IRC server called "IRCNet", channel #FIGUK from 9:00pm UT.

Until next time, sally Forth,

Graeme Dunbar



Graeme Dunbar [ g.r.a.dunbar@rgu.ac.uk ]

# Forth News

## Graeme Dunbar

A roundup of news and events from around the Forth world.

## Commercial Systems

### *VFX v3.6 for Windows*

MPE's VFX Forth for Windows development system is now at version 3.60. This new version includes DLL generation, the PowerNet Telnet and web server with CGI and ASP. The VFX code generator has been enhanced with optional Pentium 4 specific optimisations. VFX Forth can be used with Windows 98, NT, 2000, ME and XP.

http://www.mpeltd.demon.co.uk/

### *Embedded Newsletter*

Computer Solutions Ltd, suppliers of embedded systems and Forth products have released their December Embedded Newsletter for download from their web site.

http://www.computer-solutions.co.uk

## Forth Resources

### *cfdos*

cfdos is a DOS program to display colorForth source and create bootable colorForth floppy disks. A new EuroForth edition is available for download.

http://www.inventio.co.uk/

### *kForth*

An update of the Forth Scientific Library interface file, fsl-util.4th contains a bug fix and an additional definition. The following FSL packages have also been ported to kForth:

- gaussj.4th, a package for solving a linear system of equations using Gauss-Jordan elimination.
- shellsrt.4th, a sorting utility for floating point arrays.
- regfalsi.4th, a function root-finder.

http://ccreweb.org/software/kforth/kforth.html

### *F11-UK*

The F11-UK 68HC11-based Forth embedded computer developed by Jeremy Fowell with input from other members of the Group is currently out of stock. A new version is currently under development.

http://www.figuk.plus.com/f11uk.htm

Chris Jakeman [ cjakeman@bigfoot.com ]

# AGM Report

**Chris Jakeman**

The FIG UK AGM took place on Saturday 25th October at Doug Neale's home. Thanks go to Doug and to Mrs. Neale for their hospitality. Chris reports on the proceedings.

## Membership

Membership remains steady with new members replacing the lapsed ones. Jeremy's company Cobalt Controls will take out a Corporate Membership. Chris to chase MPE for payment (£180) promised in July. Chris to ask Peter Knaggs (Bournemouth University) to take out a subscription for the university.

## Forthwrite

All very grateful to Graeme for taking it on. We should respond to his appeal for material/ideas/participation to make the magazine a club effort. I now have a new job, as a lecturer in computing. It pays less than half my previous salary but it's great to have a secure future again (and the summer holiday will be good too). I've still working all hours though, learning how to teach and preparing for courses where the material is in chaos or just missing.

## Library

Graeme reported the borrowing records have been lost. Recommend that he asks through Forthwrite for members with books on loan to get in touch. If any of the few books still in print have gone missing, he should buy replacements.

Doug recommends two web sites for used books at reduced prices - www.half.com and www.abebooks.co.uk. Jenny, can you investigate prices for "Starting Forth"? If it looks promising, can we have a recommendation on the web site about it?

## F11 UK

Jeremy has 2 more customers for the board and his stock of 30 is exhausted. (Enquiries are running at 2/month). He is partway through a re-design (50 off) which will be better but cost no more. He's promised to write about the proposed improvements in Forthwrite.

There will no longer be a special discount for FIG UK members (£2) as this causes more trouble than it's worth.

## www.fig-uk.org

Chris to resolve the domain issue which forces us to use re-direction on the web site (and damages our ranking on Google).

## Bulletin Board

Jeremy and Doug agree with me that a bulletin board, hosted on the www.fig-uk.org site, would be a good way to encourage participation and community for FIG UK. Some people are discouraged from posting on comp.lang.forth because of its global scope. Web-based bulletin boards are slick, cheap (or free) and can be split into forums, such as FAQ, F11 UK, etc.. They can remember what you have already read and highlight the new, unread material. They can also be configured so that all visitors can read, but only FIG members can post. Also you can be notified by email when a forum changes (with a link to click that takes you straight into the forum).

The committee asks Jenny to do some research on this, and let it know what she thinks.

## 25th Anniversary

Jeremy has offered hospitality for this event in Nov 2004, especially as Birmingham is more central than London. We've plenty of time to think about this, so any and all suggestions welcome.

## CD

A useful CD will be something physical and convenient we can offer new members (incentive to join) and offer on the web-site to non-members as another reason to visit. Doug has lots of ideas for this. It might contain: - Forthwrite - Charles Moore videos - MPE material (supporting our UK Forth industry) - public domain products and documentation.

Doug will negotiate with Stephen Pelc at MPE for the right to disseminate MPE material and will burn the CDs. Chris to discuss the publication of Charles Moore videos with Jeff Fox. We will have to charge non-members something to cover our costs. (They offer a CD already for £25 - see the MPE advert on the back cover of Forthwrite).

The committee asks Jenny to investigate the use of PayPal to take payments. I've done something on this in the past but it's some time ago now.

# *EuroForth 2003 – The Report*

## *Howerd Oakford*

Howerd Oakford of www.inventio.co.uk reports on this year's EuroForth Conference, held on the 17th to19th of October 2003.

This year's EuroForth took place in Ross-on-Wye, UK, just on the English side of the border between England and Wales. The bar of the Royal Hotel gives a magnificent view over the Wye valley, and was the scene of many lively discussions, likewise the comfortable lounge area. Particularly notable was the flock/herd/squadron (what is the collective noun?) of floor robots performing synchronised dance moves, programmed of course, in Forth.

Numbers were down on previous years – a total of sixteen people from eight countries – six from academic institutions, ten commercial. The small numbers, quiet, comfortable hotel and relaxed atmosphere made this EuroForth particularly special for me – more of a long weekend party than a computer conference, and for this we owe Janet and Nick Nelson of Micross (http://www.micross.co.uk/) for their splendid "cavalier" organisation. Thanks!"

Between the sumptuous meals, the workshops held in and around the Jacuzzi and sauna of the local health spa, the walk up the hill overlooking Ross and the trip to Symonds Yat tourist spot, there were some good papers presented:

**Program Analysis for Stack Based Languages – Jaanus Pöial** – presented two alternative methods for formally analysing the stack effect of a Forth (or other stack-based language) program: "must" and "may" analysis. The less stringent "must" analysis requires that the stack effect of branches must not be affected by whether the branch is taken or not, and loops must not change the stack on each iteration. By imposing these restrictions the formal proof of the correctness of the program becomes much simpler, but nonetheless provides useful practical information.

This paper and the one following are part of the study of formal methods applied to Forth – because Forth is simple enough to study in this way.

**Using Forth in an Investigation into Reversible Computation – Bill Stoddart** – gave some fascinating insights into the theoretical lower limit of energy required to measure or change the state of single bit. This is important because if you can make every step in a computation reversible you can approach its maximum efficiency. Again it is Forth's simplicity that allows it to be modified to run reversibly, to create a platform to put theory into practice. [Following discussions in the bar, I found out that you cannot use a reversible program to calculate a non-reversible function - so my dream of reversing RSA on my PC is shattered!]

Bill presented a paper that showed that computing can be made "logically reversible", and described two implementations of reversible Forth machines. This has profound implications for the efficiency of real processors as the energy used to change each bit of data gets small enough to be influenced by quantum effects. Fascinating …

**Implementation Issues for Superinstructions in Gforth – M. Anton Ertl & David Gregg**. A “superinstruction” is an assembler macro that implements a group of Forth words. By making the compiler recognise these groups of words the compiled code can be made to run two to three times faster. There are many different trade-offs between compiler complexity, program size and speed which depend on the processor architecture. Anton and David have investigated the optimum set of superinstructions to use on a variety of processors, and the causes of each type of improvement in efficiency. This research has a direct effect on the speed of Gforth, one of the new native-compiling Forths which are allowing Forth to shed its “interpreted = slow” label.

**A hydraulic servo controller using the IX1 microprocessor – A.C. Wheatley & N.J. Nelson**. For this “paper” Nick escorted us all to Travers Metal Products in nearby Coleford to see Forth in action. Those of you who were at EuroForth 2000 will certainly remember Nick’s graphic demonstration of a 10 axis metal bending machine – now we saw the real thing. Modern cars are designed with extremely complex curves, and we witnessed a 1.5 m length of straight mild steel “C” section being turned into a curved door surround.

Thanks to Steve Richards of TMP for the detailed explanation of the stretch forming process : the ends of the metal are gripped by two turntables which can be moved apart as they rotate, bending and stretching the metal around specially made formers. The tension applied to the metal must be calibrated by using piano wire so that corners are formed without rippling the inside of the bend, and so that the correct shape will be made when the tension is released. Forth has been an integral part of the development of the system, both in the control hardware (an IX1 Forth chip) and also the interactive environment running on a PC.

**The colorForth Magenta Variable – Howerd Oakford**. One of the innovations in colorForth is the use of “pre-parsed” source code. Rather than store the text that you type directly into the source block, the characters are compressed into 32 bit “tokens”, 4 bits of which are reserved for “colour”. When the editor displays each 32 bit token it selects which colour to use, and when this source is loaded these colour bits control the compilation process: red creates a new definition, white is a comment, yellow immediate, green is compiled and cyan searches the “macro” vocabulary. This has several effects:

- There is now no compilation state (or STATE variable) so words that don’t end in a semi-colon carry on executing the next word.
- Names are all 32 bits long (with extensions for text comments) , so the dictionary is an array rather than a linked list.
- The name of the word being typed can be held in the top stack item.
- The editor can display each colour token in a different way.

It is the last effect that creates the possibility of the Magenta variable, where the variable’s data is stored in the pre-parsed source. The editor displays a magenta token’s name (coloured magenta, of course), then displays the next 32 bit cell as a number – the value stored in the variable. When the magenta variable token is loaded it compiles the literal address that points into the pre-parsed source. In this paper I describe why this is important…

**Three Forths Make a Hole – Stephen Pelc and Howerd Oakford**. Every few years a project comes along that stands out from all the others – the Abrasive Cutting Equipment (ACE) project is one such project. Everything about ACE was larger than life – all customers are demanding, but Armed Forces Bomb Disposal Officers take this to new heights, they really would prefer it if the 500 Bar pneumatically controlled water jet did not crash into the 1000 pound bomb that they are trying to defuse. This is one program that can really crash with a bang!

All projects are required to be finished by the end of the day, if not hour, on which the customer decides what they actually want – ACE was unusual in that the final specification was only decided *after* the first on-site demonstration of the finished system.

Every customer wants value for money, and they would rather have the project completed in three months than six. – ACE took the phrase "tight timescale" to the level of a physical and emotional challenge.

What is clear to both Stephen and myself is that the ACE project would not have happened without Forth. In the five months between September 2002 and February 2003 (not counting a six week break over Christmas), Stephen and I programmed a graphical user interface on a PC, two embedded ARM systems, an HDLC based comms protocol (with re-tries) and four cutting speed control algorithms. We also wrote full documentation for all of the Forth code (embedded into the source with DOCGEN) and a user manual. And that was the easy bit: with the help of the other members of the team, we debugged, re-designed and repaired the hardware, mechanics and plumbing of a complex safety critical system.

One year on, and I still cannot decide whether or not the ACE project was a good or bad experience for me, but what I can say is that it would have been worse without Forth, because it would not have worked. ACE is now a viable project, thanks to Forth.

**A Framework for Literate Programming – Federico de Ceballos** - deservedly won the prize for best paper : a combination of new developments in structuring and documenting Forth programs, classic computer wisdom from Donald E. Knuth and some elegant Forth code. Like both Holon and colorForth, Federico's Literate Programming Framework gives the editor more work to do, this time to provide a hierarchical structure. Like MPE's DOCGEN system, the code and documentation is stored in the same place, making synchronisation simple.

Federico compares the conventional methods of organising Forth source and documentation – screens collected together by load screens, shadow blocks etc, and describes a new approach which includes documentation, validation code, and the ability to select one of many projects stored in the same file. Future extensions include a built-in compiler and automatic display of the source on error. I look forward to seeing more of this excellent project.

**A Forth Programming System for a Coil Winding Machine – Kazimierz Polecki**. Nick nelson was going to present this paper as the author could not attend, but decided not to due to lack of time, so this is my précis from the written presentation: one aspect of Forth which is usually taken for granted is the ability to edit, re-compile and execute or interpret in real time – something sadly lacking in non-interpreted batch-processed languages. What the user of the program sees is a truly friendly environment in which to program the machine – in this case to wind coils. This is classic Forth: a language to describe the business of coil winding, an integrated editor/compiler/interpreter that can display and/or execute the coil specification written in that language. Direct communication with the machine is so much better than the dumbed-down clicking of generic Windows programs. It's good to see that the art of programming has not been lost!

**The Industrial presentations**: **Duncan Louttit** of Swallow Systems demonstrated his Pip and Pixie range of floor robots. The same robots that had kept us entertained in the bar the previous evening were now put through their paces presenting a fashion show. Duncan showed us videos of them in action in a primary school – taking on the serious task of educating 5 to 12 year olds in the skills of logical thought and having fun. Utterly charming!

**Stephen Pelc** of MPE demonstrated the new PowerNet 3.0 Forth-based Web server, which comes packaged with the latest VFX Forth for Windows. There are two significant advances here: the fact that a full HTTP/TCP/IP implementation is now supplied free with VFX, and the existence of a Web server which uses Forth as its scripting language. The familiar "OK" prompt of the Forth QUIT loop is now available on the Web. This may sound like Telnet into a Bash shell – the difference is that you can run any Forth word that your system contains, provided you are given access, without running into the limits imposed by the Unix/Linux shell environment. This is perhaps too powerful, and some time was spent discussing the security implications of allowing anyone on the Web to open a Forth interpreter on your PC…

To end this report I must say thanks again to Nick and Janet for making EuroForth 2003 happen, to Chuck Moore for discovering and passing on Forth, and to all the friends old and new for some great conversations – covering everything from cavaliers and roundheads to Arthur Lee, with lots of techie Forth talk too. See you all next time!

Howerd 8^)

A selection of papers from the conference are available at:
http://dec.bournemouth.ac.uk/forth/euro/ef03.html - *Editor*.

James Boyd [ JimBoyd@techemail.com ]

# A Virtual Nondeterministic Machine in Forth

**James A. Boyd**

In the first part of his article, James Boyd describes a technique that has applications in search-type problems. The concluding part will be published in the next issue.

A nondeterministic machine is a fascinating theoretical construct with the ability to choose a "correct" value for each element of a solution to a given problem if a solution to the problem exists. Algorithms for a nondeterministic machine can be considerably faster than deterministic algorithms. This article describes implementing a virtual nondeterministic machine, an experiment in Forth with hopefully some of the potential of the theoretical machine. This work was motivated by a discussion of nondeterministic algorithms by Ellis Horowitz and Sartaj Sahni [1] and an article on nondeterministic control words in Forth by L. L. Odette [2].

## Theoretical Framework

Horowitz and Sahni introduced the concept of a nondeterministic algorithm in a discussion of NP-hard and NP-complete problems. They introduced a machine which only exists in theory to provide "strong intuitive reasons" to conclude that "fast" deterministic algorithms cannot solve problems in NP due to the extreme time required.

The best known algorithms for many of the problems known to computer science have computing times which cluster into two groups. Problems in the first group have a time to solution which is bounded by a polynomial function of small degree. They are said to be in P, the group of all problems solvable in polynomial time. These problems are considered tractable, solvable in a reasonable amount of time even for very large problems. Some examples include matrix multiplication, ordered searching, polynomial evaluation, and sorting.

### What is a Nondeterministic Machine?

A nondeterministic machine simply has the ability to correctly guess the correct choices to solve a problem if such choices exist. A nondeterministic algorithm is one that uses nondeterministic operations of a nondeterministic machine. For example, suppose there is a word `?even` which returns a true flag if there is an even number on the stack. In the word `evenDemo`, `choice` always chooses an even number from the range 0 to n:

```
: evenDemo    ( n -- )
        choice   dup ?even
        if       success  .
        else     failure  drop
        then ;
```

and in the word `oddDemo`, `choice` always chooses an odd number from the range 0 to n if n > 0:

```
: oddDemo    ( n -- )
        choice   dup ?even
        if       failure drop
        else     success  .
        then ;
```

The best known algorithms for problems in the other group have times to solution which are not bound by a polynomial of any size and are referred to as NP problems. NP problems are considered intractable typically with an exponential time to solution. While solvable in theory, the time grows so rapidly as problem size increases that it is not feasible to solve moderate to large problems in this group.

There is a subclass of NP problems called NP-complete. All problems in this class are solvable in theory and if any NP problem can be solved in polynomial time then all NP-complete problems can be solved in polynomial time. So far no one has been able to devise a deterministic polynomial algorithm for any NP problem. Some examples of NP problems include the travelling salesperson, the knapsack, Hamiltonian circuit, and possibly the N queens problem.

A nondeterministic machine can, in theory, find a solution to an NP-complete problem in polynomial time. A nondeterministic algorithm can be written for an NP-complete problem easier than its deterministic counterpart and the nondeterministic algorithm's time to completion is usually a polynomial of small degree. A nondeterministic machine has the operations `choice`, `failure`, and `success` [1]. The virtual nondeterministic machine has the same three operations. The code for the virtual nondeterministic machine is given in the file **VNM.F**.

Given a range, `choice` randomly chooses a number within that range such that `success` will be executed and `failure` avoided if such a choice exists.

- `choice` removes a number **n** from the stack and returns a number in the range zero to **n** inclusive.
- The phrase `over - choice +` removes two numbers **lo** and **hi** from the stack and returns a number in the range **lo** to **hi** inclusive.

`failure` is placed after a test of the choice so that `failure` is executed only if the test fails. The following code fragment shows how actions to be taken upon failure are performed:

- `... if   failure  word1 ... wordn   then ...`

`word1` through `wordn` are only executed when `failure` executes. They could do anything from cleaning up the stack and displaying an error message to exiting the algorithm. Notice that they are in the same `if ... then` phrase as `failure` and they are placed after `failure`.

`success` is placed at the end of the algorithm such that it is executed only if the problem is solvable. The one time the choice might not avoid `failure` even when there is a valid solution is when `success` is executed before `failure`. The nondeterministic behavior of `choice`, its ability to make a choice which avoids execution of `failure`, is limited to the code between itself and the first occurrence of `success` or `failure`. This is why `success` is normally placed at the end of an algorithm.

A nondeterministic algorithm is written by giving `choice` the range of possible values for each element of the “solution vector” and testing its choice for validity branching to avoid or execute `failure` accordingly. The behavior of a nondeterministic algorithm is determined by the range of possible values for each element of the solution vector and the tests or predicates[2] as Odette called them. Writing a nondeterministic algorithm is not hindered by considerations of backtracking or what to do if an element fails the test. If any choice leads to the execution of `failure` there is no solution.

Unlike the theoretical machine of Horowitz and Sahni `success` and `failure` do not terminate the program, just the nondeterministic behavior of the algorithm. They can be used in words called by the main algorithm such as the predicates, but such words can only be used in nondeterministic algorithms. Since `success` and `failure` end the nondeterministic behaviour of the algorithm, `success` can be placed at the beginning of a nondeterministic algorithm to isolate it from a nondeterministic algorithm which has not been properly terminated with `success` or `failure`.

## *Queens Problem*

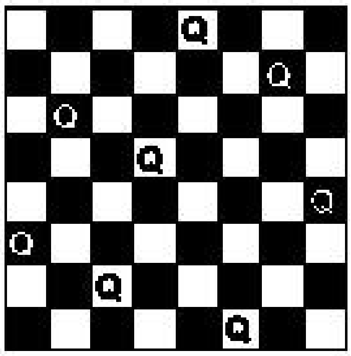


***Figure 1***

Consider the eight queens problem as an example. The goal is to place eight queens on a regular chessboard such that no queen is under attack. The file **NQUEENS.F** is the code for `nqueens` the nondeterministic algorithm for the N queens problem. Notice that `success` occurs at the beginning and end of the algorithm while `choice` and `failure` are in the subroutine `(nqueens)` which does the actual work. Also notice the predicate `attacks?` is right after `choice` to confirm the validity of its choice. ***Figure 1*** shows one solution to the eight queens problem. The solution is represented as a list of row positions for each queen: 3 6 2 5 8 1 7 4. The solution in ***figure 1*** is not the first one a conventional algorithm would return. There are 92 solutions to the eight queens problem and a nondeterministic algorithm could return any one of them.

A conventional backtracking algorithm needs to try each value for each element of the solution until a position is found where the queen being placed is not under attack. If there are no positions where the queen is safe then the algorithm must backtrack to the previous column and resume placing that queen by trying the positions farther than its current one. If the previous queen can not be placed then the algorithm must backtrack to the queen before that one. All this backtracking slows the algorithm down a great deal. If each "try" is the selection of a new position and testing the queen's placement then the eight queens problem requires 876 tries to place all eight queens.

Given a nondeterministic machine's ability to choose the correct value for each element of a solution, its algorithms need no backtracking. For the eight queens problem the nondeterministic machine requires only eight tries to place all eight queens.

The eight queens problem can be generalized to the N queens problem. The goal is to place N queens on an N by N chessboard. A nondeterministic machine requires only N tries to place all

N queens. Since there is no solution for the two queens and three queens problems, a nondeterministic algorithm would terminate with an error message if it tried to solve one of these problems and the output, if printed, would be meaningless.

## References

[1] Fundamentals of Computer Algorithms by Ellis Horowitz and Sartaj Sahni

[2] Nondeterministic Control Words in Forth by L. L. Odette Dr. Dobbs Journal September 1983

***James A. Boyd***

I am a Christian, a Husband and Father of four. Although not a professional programmer, I have programmed in Forth as a hobby since November 1991.

I became interested in nondeterministic machines when I read L. L. Odette's article and a book titled "Fundamentals of Computer Algorithms". In the chapter on NP-hard and NP-complete problems, Ellis Horowitz and Sartaj Sahni describe a theoretical machine called a nondeterministic machine.

*Editor's note: the concluding part of this article will be published in the next issue. In it, matters of implementation and usage are discussed and a number of test results are presented. The author recommends reading both parts of the article before trying the code.*

## File: VNM.F (sheet 1 of 3)

This is the nondeterministic machine source. It needs other words defined in the files NONSTANDARD.F and VNMHISTORYBUFFER.F.

```
\ Virtual nondeterministic machine
\ James Boyd    October 12th, 2003 - 5:04
\ needs words from just about all over the standard plus a few that aren't
\ see file NONSTANDARD.F

decimal
needs vnmhistorybuffer.f

\ *********************** Stacks ******************************************
comment:
        The words used to save and restore the stacks. The idea to save and
restore stacks came from L. L. Odette's article. This fast version is
specifically for WIN32FORTH. See file VNMSTACKS.F for other stack saving and
restoring words.
comment;

: SaveReturn   ( -- )
        rp@  cell+  rp0 @ over - tuck  ( # adr # )
        m>history  >history ;

: RestoreReturn   ( -- )
        r>  ( naddr ) \  nesting address
        history> rp0 @ over -  rp!  ( naddr # )
        rp@ swap mhistory>  ( naddr )
        >r ;

: SaveDataStack   ( -- )
        sp@ sp0 @ over - tuck  ( # adr # )
        m>history  >history ;

: RestoreDataStack   ( -- ? )
        sp0 @ history> dup>r  ( adr # )
        - sp! ( ? )
        sp@ r>  mhistory> ;

comment:
        These defered words allow adding code to save and restore other data
with the machine state
        Just remember, whatever is saved to the history stack must be restored
in the reverse order.
comment;

defer SaveOther         ' noop is SaveOther
defer RestoreOther      ' noop is RestoreOther

: SaveData   ( -- )   SaveDataStack  SaveOther ;

: RestoreData   ( -- )   RestoreOther  RestoreDataStack ;
```

## File: VNM.F (sheet 2 of 3)

```
\ ************************* VNM primitives **********************************
comment:
        choice# is a defered function generator. the generator MUST return a
number in the range zero to N-1 where N is the number supplied to the generator.
The number generator does NOT HAVE to be a random number generator.
        Safer function generator:
: SaferChoice#   ( n -- n2 )   dup choice# swap 1- umin ;

        choice# is NEVER called with a parameter of zero.
comment;


defer choice#           ' random is choice#

\ Generate the group data in compressed form with one choice removed

: group   ( n -- n2 )
        dup 1+  choice#
        2dup >history >history  2 >history  - ;

\ Randomly select one choice and remove from the group.

: (choice)   ( -- n )
        pswitch  history> dup 2 u<      \ ( size f )
        if   drop  history>  pswitch  exit   then
        history>  main>aux  over        \ ( size last_element size )
        choice# pointer-                \ ( size last_element )
        1 pointer+ history> -rot        \ ( chosen_elemen size last_element )
        >history pswitch  1- >history ; \ stuff last where chosen_element was

: generate   ( n -- ) \ Generate the set or group from compressed form.
        dup cell  over
        if   um*   then   ?history      \ room for entire group?
        0  do   i (>history)   loop ;   \ generate group fast, room is available

: expand   ( -- n ) \ expand the rest of the group data with data from history
        ['] (choice) >history   \ replace top of saved ret stack with (choice)
        1 pointer-              \ restore pointer to where it was
        pswitch  history> drop  \ discard top of history stack -- it is 2
        history> history>  over \ ( last_element displacement last_element )
        1+ generate             \ ( last_element displacement )
        tuck  1+ pointer-       \ back pointer to where first chosen value is
        >history  pointer+      \ stuff last where first chosen value is
        pswitch (choice) ;
```

## File: VNM.F (sheet 3 of 3)

```
\ ************************ Main VNM words *********************************
comment:
        Most nondeterministic algorithms can be written with just the three
nondeterministic operations choice success and failure.
comment;


: choice   ( n -- n2 )
        depth 1 < abort" Unspecified set"
        dup if  ['] expand >r  SaveReturn  r> drop
                >r SaveData r>  group ( n2 )
        then ;

: success   ( -- )
        ['] noop dup is SaveOther  is RestoreOther  remove ;
\ Failure primitive -- will not display error message if history data exausted
: (failure)   ( -- ? ) \ if machine state not restored then no stack effect
        exhausted?                            \ if any history left
        0= if   main>aux  history> pointer-     \ skip over group data
                RestoreData  RestoreReturn      \ restore machine state
                r> execute                      \ and execute cfa left on
                                                \ return stack
        else    success                         \ else free memory for history
        then ;

: failure   ( -- ? ) \ if machine state not restored then no stack effect
        (failure)
        cr ." No Solution" ; \ if there is history then this will not execute


\ ************************ suspend utility ********************************
comment:
        A useful "utility" word, suspend saves history data and aborts so an
algorithm can be "suspended" checked out and then resumed by typing failure.
        upon failure a no-op is executed and the machine state is restored to
where it would be if suspend had not occured.
comment;

: suspend
        ['] noop >r  SaveReturn
        SaveData
        0 >history                      \ no group so save zero group size
        cr ." Type failure to resume or"
        cr ." Type success to clear history data"
        cr .s
        abort ;
```

## File: VNMHISTORYBUFFER.F (sheet 1 of 2)

Words needed by VNM.F.

```
\ History buffer implemented with heap memory
\ James Boyd    October 31st, 2003 - 19:15

Comment:
        This implementation uses the Standard Forth words
ALLOCATE FREE & RESIZE
Comment;

\ The history stack/buffer is a lifo structure for the most part.

decimal
0 value history         \ when non-zero, it holds location of history stack
0 value historySize
variable MainPointer    MainPointer off
variable AuxPointer     MainPointer off
cell value pagesize     \ actual size made little difference in nqueens times


: main>aux   ( -- )   MainPointer @  AuxPointer ! ;

: pswitch   ( -- ) \ switch main and auxilliary pointers
        AuxPointer @  main>aux  MainPointer ! ;

: remove   ( -- ) \ zero both pointers and free allocated memory
        MainPointer off  main>aux
        history if
                history free drop
        then
        0 to history  0 to historySize ;

: ?history   ( d -- ) \ make sure there is room for d bytes
        MainPointer @ 0  d+             \ ( d2 )
        2dup historySize 0  d<          \ ( d2 flag )
        if   2drop exit   then
        pagesize 0  d+                  \ ( d2 )
        dup if  remove  then            \ clear history and
        abort" History Full"            \ abort on overflow
        history  ?dup                   \ ( n addr addr or n 0 )
        if      over resize             \ ( n addr2 ior )
        else    dup allocate            \ ( n addr2 ior )
        then
        dup if  remove  then            \ clear history and
        abort" History Full"            \ abort if no more memory  ( n addr2 )
        to history  to historySize ;

: exhausted?   ( -- f )   MainPointer @ 0= ;

: pointer-   ( n -- )   cells negate  MainPointer +! ;

: pointer+   ( n -- )   cells  MainPointer +! ;
```

## File: VNMHISTORYBUFFER.F (sheet 2 of 2)

```
: (>history)   ( n -- ) \ save cell 'n' to the history stack
        MainPointer @ history + !
        1 pointer+ ;

: >history   ( n -- ) \ save cell 'n' but check to make sure there is room
        cell 0 ?history  (>history) ;

: history>   ( -- n ) \ retrieve cell 'n' from history stack
        1 pointer-
        MainPointer @ history + @ ;

: m>history  ( addr n2 -- ) \ save n2 bytes to history from addr
        dup 0 ?history
        MainPointer @  history + swap dup
        MainPointer +! cmove ;

: mhistory>   ( addr n2 -- ) \ transfer n2 bytes from history to addr
        dup negate MainPointer +!
        MainPointer @  history + -rot cmove ;

remove
```

## File: NONSTANDARD.F

Non-standard words used by VNM.F.

```
\ Win32Forth words used which are not standard

\ dup>r   just replace dup>r with  dup >r

: -rot   ( n1 n2 n3 -- n3 n1 n2 )   rot rot ;

1 cells constant cell

: noop ;

: off   ( addr -- )   false swap ! ;

: ?cr   ( n -- )   drop ;        \ Don't know how to do this one, sorry.

\ Code for IS borrowed from Rick VanNorman's 32 bit Forth for its portability

: is
   state @ if    ' >body postpone literal postpone !
           else  ' >body !
           then ; immediate

: defer   create  ['] noop ,
   does>   @ execute ;

\ 31 bit pseudorandom shift register in software
variable rseed  here rseed !

: random   ( -- )   rseed @  dup
        16 lshift   swap
        2* dup   3 lshift   xor   16 rshift
        or dup rseed !
        16 lshift  um*  nip ;
```

## File: NQUEENS.F

This is the NQueens problem, given as an example demonstrating the use of the program VNM.F. The behaviour and performance of the software will be discussed in the next issue.

```
\ N queens problem
\ James Boyd    September 22nd, 2003 - 3:30

decimal
0 value queens
: freeQueens   ( -- )
        queens if   queens free drop   then
        0 to queens ;
: setQueenSize   ( n -- )
        freeQueens
        cells allocate   abort" No room for queens array"
        to queens ;
: attacks?   ( column queen -- column queen f )
        key?
        if   suspend   then
        over cells queens +   queens            \ if two queens on same row or
        ?do     i @  over = ?dup                \ diagonal we exit with true
                if      unloop exit   then      \ flag -- the new queen is under
                over  i queens -  cell /  -     \ attack.
                over  i @ -  abs = ?dup         \ otherwise we exit with a
                if      unloop exit   then      \ false flag -- no attack
        cell +loop
        false ;
: addqueen   ( column queen -- column+1 )
        over cells  queens + !  1+ ;

: showqueens   ( n -- )
        dup  cr space 3 .r ."  queens" cr
        0 do    i cells queens +  @ 1+  space 3 .r  16 ?cr
        loop ;
: (nqueens)   ( n -- )
        ?dup 0= if   cr ." No solution for zero queens" exit   then
        dup setQueenSize
        1- 0    ( n-1 column )
        begin
                over choice attacks?   ( n-1 column queen f )
                if      failure  2drop drop
                        exit
                then
                addqueen  2dup <        ( n-1 column f )
        until   ( n-1 n )
        space showqueens   drop ;
: nqueens   ( n -- )   success (nqueens) success ;
```

# Library Notes

**Graeme Dunbar**

News and Comment from the FIG UK Librarian.

## Relocation Problems!

In the last issue I reported that the FIG UK Library was in the process of being moved to new accommodation due to building work within the School of Engineering at Robert Gordon University in Aberdeen.

The good news is that it has been unpacked and largely re-shelved and there is a little more space to organize the material in a logical fashion. However, the bad news is that part of it has gone missing. All the journals (back issues of Forthwrite and Forth dimensions), the conference proceedings and the FIG implementation listings are accounted for. As far as I can tell none of the books have vanished. That would suggest that the library is intact – but – I said “as far as I can tell”. The index card catalogue and all the records of past and present loans have vanished. In addition, the copy of the ANS Forth standard and other material printed out several years ago on fan-fold computer paper have gone.

Unfortunately the room where the Library used to be shelved was requisitioned by another part of the university at short notice and was cleared when I was on holiday. Apparently the missing material was not packed into banker’s boxes or library “coffins” with the rest of the material and was subsequently misplaced. There is a slight chance that it will turn up in a box with some totally unrelated material, but the most realistic explanation is that it ended up being shredded.

## Loans

If you have any material from the FIG UK Library in your possession, then please let me know what it is as soon as possible. I do not need it back at the moment, but I do need to trace all material on loan. It would be helpful if you could give me the full bibliographic details (e.g. author surname and forename or initials, title, edition, publisher, publication date and ISBN) as I have to rebuild the catalogue. If you wish to borrow any material, I will do my best to get it to you as soon as possible.

## Catalogue

The old catalogue was hand-written on 5” x 3” index cards and held a record of borrowers going right back to the first days of FIG UK. It is as much a shame that a bit of history has been lost as is the inconvenience and effort required to produce a new catalogue. The plan is to produce the new catalogue on a computer, possibly using Microsoft Access. This is a considerable task and will take several months to complete.

Henry Vinerts [ Volvovid@aol.com ]

# 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 – September 2003

Hello friends,

Dr. Ting was missing from our September meeting, and fewer than a dozen of us met without a planned full-day schedule, which was probably somewhat disappointing to our surprise visitor -- Bernd Paysan of Munich. To our delight, Bernd filled in a large part of the day with authentic information about GForth updates and miscellaneous news from the other side of the "Big Pond." Many thanks to Bernd!

I believe that Bernd also made it to the Vintage Computer Festival on October 11th or 12th, where SVFIG too had a booth among some 30 other exhibitors. The festival was staged by the Computer History Museum in a new and impressive facility in Mountain View, CA, that used to belong to Silicon Graphics. I wrote about the origin of the museum last year when I visited it still at the NASA facility in Sunnyvale. There was some repetition of exhibits from last year, including Hans Franke's (from http://www.gfhr.de ), and I must remark that "vintage" does not mean "antique". There were few items older than the MITS Altair. As for what the museum itself offers to the public, perhaps I should just point out that one can visit http://www.computerhistory.org/VirtualVisibleStorage/ on the Web.

Also, I happen to have received an interesting news article through the National Society of Professional Engineers, which I am copying below. Perhaps, with some condensing, there may be material of interest to your readers.

**Technologists Predict Bold Innovations for Valley SiliconValley.com (10/13/03); Langberg, Mike**

"Future technological advancements that could transform society and raise Silicon Valley up from its economic doldrums were the subject of the Silicon Valley 4.0 conference at the Computer History Museum in Mountain View, Calif., on Oct. 7. Innovations predicted at the conference included ubiquitous wireless: SRI International think tank president Curt Carlson proclaimed that concurrent breakthroughs in speech recognition, smooth networking, and longer-life batteries will yield mobile devices that will become as essential to people as wallets, while wireless analyst Gerry Purdy forecast that future devices will be compatible with all available wireless networks and capable of automatically retrieving desired information. Speech recognition will prove vital because people are uncomfortable with miniature keypads on handhelds. Entrepreneur and Packet Design Chairman Judy Estrin anticipated the launch of a "phenomenal cycle" with the advent of embedded computers in approximately five years. She also expects the creation of massive networks that will monitor traffic, the environment, and many other things when embedded computers and sensors are linked. Estrin explained that Silicon Valley can only meet this challenge by developing a way to design small, low-power computers. Jeff Hawkins of Handspring reported that neuroscience researchers are close to understanding how the brain creates and comprehends human speech, and added that the replication of these processes in silicon chips will spur a "totally unexpected" computing revolution. Biological engineering was also a topic of discussion, with Geoffrey Moore of Mohr Davidow Ventures noting that his company is investing in businesses focused on the design of medical monitoring chips."

(http://www.siliconvalley.com/mld/siliconvalley/7001991.htm)

This concludes my contribution for now. I hope to have some more after the next SVFIG meeting, which is this coming Saturday.

Glueckauf,

Henry

## FIG Silicon Valley Chapter Meeting – October/November 2003

Hello, everybody!

In case you are wondering, I have not written about the SVFIG meetings of October and November, because I waited until after Donald Knuth's lecture at the Computer History Museum to put my reports all together in one package and to close with best wishes to all in the fast-approaching year 2004.

On October 23rd our usual core group of about fifteen--plus/minus three--members spent the morning listening to Dr. Ting's description of a new "souped-up 8051" board developed in Taiwan and how eP32, eForth, F#, an RGB display, FML, etc, may make this the "Chinese computer-to-be." The afternoon saw Dwight Elvey's slide show from the recent Vintage Computer Fair and some tips on how to boot really old computers and save and recover old software.

The yearly "Forth Day" came on November 22nd, and I must say that it was a pleasant surprise to count nearly 30 people in attendance at the very start. Kevin Appert had done a superb job in locating, contacting, and inviting many forthers whom we had not seen for a long time. The disappointment was that this may have been the first year that Chuck Moore did not come to entertain us with his "fireside chat." He and his wife moved to Sierra City (elevation 4500 feet) in the California Sierras a little over a year ago. Jeff Fox, who has gone up the mountain, reported difficulties in keeping up with the Moores when they all went hiking in the snow.

I am sure that the SVFIG web page shows the schedule and the topics of the talks of the eight presenters, all of whom seemed to have more to offer than time permitted, therefore I'll restrict myself to a few items of non-technical interest.

The slightly cold wind did not prevent Ting's traditional outdoor barbecue lunch from being a successful socializing event. The new faces among the presenters were Randy Thelen with his home-made TTL board running Mippy ("million instructions per year"), and Joseph M. O'Connor, who had come across the country from Connecticut to present Creole, a Forth-like scripting language in Borland Delphi. La Farr Stuart returned to meet some of his friends from the 1970s and to amuse us with Forth in mathematics. Jay Melvin claimed to have been the first customer of Tom Zimmer's and have named the purchase ZForth. Tom was one of the old forthers who could not come but had responded to Kevin's invitation with an e-mail letter. The only complaint of the day that I can register is that the fonts in the Power Point presentations are not friendly to our group which grows increasingly presbyopic by the year.

The Computer History Museum seems to be doing well in its new location in Mountain View, California. Even if I repeat myself, it is worth visiting www.computerhistory.org to learn more about its exhibits, lecture series, and various activities.

In the book by Shasha and Lazere "Out of their Minds - The Lives and Discoveries of 15 Great Computer Scientists" the following quote by Donald Knuth appears twice:

"It's not true that necessity is the only mother or father of invention. The other part is that a person has to have the right background for the problem. I don't just go around working on every problem that I see. The ones I solve, I say, 'Oh, man, I have a unique background that might let me solve it--it's my destiny, my responsibility.'" The chapter on Knuth ends with a statement that "Folklore holds that Knuth is the greatest computer programmer of all time." The book was written in 1995, and by now the word "programmer" doesn't even sound glorious enough to young whippersnappers who barely get through college and are given degrees in computer science. I see Knuth as a genius in mathematics, music, and computer science, all of which, as he says, have the science of patterns in common. When I look at the books that he has written, I realize that he is right, that only about 2 percent of the population have the mental qualities that resonate with computers, that qualify them as computer scientists.

Knuth was at the museum last Wednesday to give a talk on the history of computer languages and to sign his new book "Selected Papers on Computer Languages." Specifically, he described about a dozen precursors of Fortran, by showing how a simple algorithm might have been coded in each of the different languages. The point he stressed was that we did not arrive at the basic concepts of programming languages as we know them today without a lot of hard work by many brilliant and dedicated people. The list of names from Zuse to Backus and the names of the various languages that he talked about will take too much space in this letter. I understand that all of that is well covered in the new book of his. I just want to add a final note here that Knuth (and his former student and collaborator in the TPK algorithm presentation, originally in the 1970s) used color-coded syntax in the slides to help legibility of the program samples. That made me wonder whether ColorForth may some day be mentioned in the history of computer languages.

Happy New Year!

Henry

## Letters

To borrow the immortal words of Humphrey Lyttleton, we have almost two letters in our postbag. However, this one is *not* from a Mrs Trellis of North Wales . . .

Hi Graeme,

I know that Forth is a minority language and I haven't seen a Forth book on a shelf in a bookshop for years, but I have always taken some comfort from the thought that it had a couple of dedicated niche areas in which, if not supreme, it had a respected place.

One of these areas was robotics and the other was embedded systems. At the AGM our respected chairman proudly showed us his latest motor control 'block' with a Forth implementation tucked somewhere inside and I thought all was well with the world of Forth.

This was shattered this week when I received a catalogue on 'Embedded Systems' from Newnes. It proudly listed some 22 books all about embedded systems for just about every chip you could think of, together with the languages available.

NOT ONE WORD about Forth anywhere!!!!

Yours in woe,

Douglas Neale

Joe Anderson [ jia@jia.abel.co.uk ]
phone: 0131 662 4007

# Vierte Dimension 2/2003

## Joe Anderson

Joe provides a look at the latest issue of the German FIG magazine. To borrow a copy or to arrange for a translation of an individual article, please call Joe.

### Editorial 4

Friederich Prinz

Friederich Prinz notes that Forth-Gesellschaft in spite of its "great age" of by now 20 years (Jubilee in 2004) has remained young. New ideas are constantly coming in, and again and again inquisitive young people are making their way into Forth-Gesellschaft as new members.

### Readers' Letters 5

Tim Daneliuk

Five readers' letters:

Correction to iSort; Comment on the reader's letter of Gerard Bäcker (issue 1/2003) about Forth and the other computing languages; Call for a programming competition; Suggestion for a programming competition; Report on SmartCards.

In addition an appeal from the Board about publishing of a list of members and the announcement that the management of Forth-Gesellschaft is no longer situated in Rostock but in Munich.

### What – a Turing Award for Moore? 8

Friederich Prinz translates a posting from the comp.lang.forth forum into German. ACM has warded the Turing Prize 2001 to Dahl and Nygaard. Tim Daneliuk, himself not a Forth devotee, asks the question why the prize should not have been awarded to Chuck Moore. He states: "If Moore had crammed his work with a load of unnecessary and obfuscating mathematics, then maybe he would have been taken seriously into consideration by the ACM."

### How a programming language is chosen. 10

Tim Daneliuk

Friederich Prinz has translated an article by Tim Daneliuk from comp.lang.python (1st May, 2002), pseudo-scientific and therefore maybe not to be taken all that seriously, but just because of its at times partly heretical thoughts very interesting. The word "Forth" doesn't appear in this article, but the author would really like once and for all to put an end to this ever-recurring question "Why is programming language X better than Y?"  Let's take one of the heretical thoughts: "Code portability is over-valued. Far more important is the portability of your technical skills as a programmer."

## *The Forth Office.* 12

Rolf Schöne

The management of Forth-Gesellschaft (the Forth Office) has moved from Rostock to Munich. Rolf Schöne introduces himself to the members. The e-mail address remains as before.

## *Advertisements.* 13

The display of the members advertisement for FIG UK and the Dutch Forth-gebruikersgroep. Additionally two photos of the "every-picture-tells-a-story type": “Civilization” (a ruined castle in disrepair) and “Rapt Attention” (people dozing at a talk). And a picture that shows the Thank-you Bear (to Ute Woitzel for five years in the Forth Office) and the SWAP-dragon (this time to Thomas Beierlein) happily together.

## *Minutes of the General Meeting in Lambrecht on 13th April 2003* 14

Michael Kalus

The budget balances. The membership count is static. Probably it has bottomed out. The previous Board (Bernd Paysan, Ulrich Hoffmann, Fred Behringer) has been re-elected. The Forth Office (management of Forth-Gesellschaft) will now be headed up by Rolf Schöne. A few projects were discussed (books, Forth processors, robots used in teaching). This year’s SWAP-dragon prize goes to Thomas Beierlein for his general work in the service of Forth-Gesellschaft. Page 15 shows a picture.

## *The FORTH Program for Spectral Line Observing.* 15

Charles H. Moore and Elizabeth D. Rather

Friederich Prinz has dug out one of the first published appearances of Forth and translated it for German readers: “The FORTH Program for Spectral Line Observing”, Proc IEEE, Vol. 61, No. 9, September 1973. What is said in this article about the advantages of Forth is still valid today and is well worth reading. Of interest are the two short biographies with very pleasing photos. The photos show how quickly time passes. The original article and both photos can be requested from the Editor.

## *Reviews.* 22

Fred Behringer

Fred discusses Forthwrite Issues 119 and 120 and Vijgeblaadje 36.

## *Clever questions, reliable answers.* 25

Friederich Prinz

Friederich has gathered a few questions and corresponding answers from comp.lang.forth.

## *Debugging with a Mixed Signal Oscilloscope with I2C-Bus as example.* 27

Klaus Zobawa

A practical application. Development of a surgical device. “Can embedded software be debugged and tested without an in-circuit-emulator?” “Yes, with Forth.” With the Field-Forth package from the Rostock Group. Implementation of a multimaster-communication over the I2C-bus. Use of the oscilloscope mentioned in the title.

***Triceps.*** **29**

Ewald Reiger

A simple robot for demonstrating pick-and-place tasks.

A paper read by the author at the Forth annual conference 2003 in Lambrecht. The model of a 3-axis machine for transporting ferrous metal parts. A magnet is used as a grab. The programme was developed with BigForth under Linux and Windows. It accesses directly the registers of the parallel port. At the moment, the programme does not lend itself to operating under Windows XP or Windows NT. Windows 95, Windows 98, and Windows ME permit execution, though here it is very slow. The speed is adequate for demonstration purposes. A trapezoidal acceleration and braking ramp ensures gentle starting and stopping of a motion.

# *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 32 Euro (£22) per year (16 Euro (11 Pound) 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@FORTH-EV.DE

or visit:

http://www.forth-ev.de/

or write to:

Forth-Gesellschaft e.V.
Postfach 19 02 25
80602 München
Germany

Telephone:

(089) 1 23 47 84

# *What Languages Fix- Not!*

The last issue of Fortwrite contained an observation on programming languages taken from the web page of Paul Graham, author and designer of the Arc language ( http://www.paulgraham.com/fix.html ).

His general thesis is that new languages are developed to fix perceived problems with existing one. The piece in the last issue ended with a question asking what can be said about Forth in this context. It was not intended to be rhetorical! So far, no one has put forward any suggestions, so the challenge is being held open until the next issue. Contributions to the Editor please.

To start the ball rolling, my own thoughts . . .

**Forth**: .................. Writing test harnesses is a pain.

I cannot think of another language where it is so easy to test as you go along.

Following on from that: there probably isn't an easier environment to write test software in either. The code can be left in with the application, ready to be used when needed. – *Editor*.

## FIG UK Contacts and Information

| | | |
|---|---|---|
| ***Chairman*** | **Jeremy Fowell,** | 11 Hitches Lane, EDGEBASTON B15 2LS<br>0121 440 1809 jeremy.fowell@btinternet.com |
| ***Secretary*** | **Doug Neale,** | 58 Woodland Way, MORDEN SM4 4DS<br>020 8542 2747 dneale@w58wmorden.demon.co.uk |
| ***Editor*** **(temporary)** | **Graeme Dunbar,** | School of Engineering, The Robert Gordon University, Schoolhill, ABERDEEN AB10 1FR<br>01224 262415 g.r.a.dunbar@rgu.ac.uk |
| ***Treasurer*** | **Neville Joseph,** | Marlowe House, Hale Road, WENDOVER HP22 6NE<br>01296 62 3167 naj@najoseph.demon.co.uk |
| ***Webmaster*** | **Jenny Brien,** | Windy Hill, Drumkeen, BALLINAMALLARD, Co. Fermanagh BT94 2HJ<br>02866 388 253 webmaster@figuk.plus.com |
| ***Librarian*** | **Graeme Dunbar,** | School of Engineering, The Robert Gordon University, Schoolhill, ABERDEEN AB10 1FR<br>01224 262415 g.r.a.dunbar@rgu.ac.uk |

Membership enquiries, renewals and changes of address to Doug.
Technical enquiries and anything for publication to Graeme.
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 Graeme Dunbar, 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. |

**Copy deadlines:**

**Issue 124: 21 January 2004**

**Issue 125: 24 March 2004**

**All material for publication to the editor by email or post by that date please. Plain text, MS Word or Rich Text format preferred.**

Back cover (Advert)