

Forth Programming

FIGUK magazine:

Start Simple

Simple State Machines

Regions: into unknown Win32Forth

Forthwrite 125 — May 2004

(inside front cover – blank)

| | |
|---|--------------------|
| | events |
| Presenting the FIG UK Awards of 2003 | 24 |
| | news |
| Forth News | 3 |
| Library Notes | 27 |
| | reviews |
| Book Review | 20 |
| Across the Big Teich | 25 |
| Vierte Dimension 4/2003 | 28 |
| | programming |
| Simple State Machines | 6 |
| Regions: into unknown Win32Forth | 10 |
| Start Simple | 21 |
| | people |
| Situations Vacant | 4 |
| Membership Matters | 5 |
| Letters | 18 |



Editorial

We have an excellent mixture of articles in this issue: something for beginners and experts alike. I encourage all FIG UK members to follow the authors' lead by sending in your contributions for publication no matter how big or small, erudite or everyday.

On the matter of writing for Forthwrite, the Committee is looking for volunteers to research and write the regular "Forth News" and "From the Net" columns. The latter has not appeared for some time and I'm afraid the News column may be cut if someone does not step forward. Please contact the editor if you are interested.

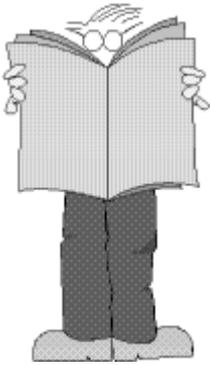
I am pleased to report that Jenny, our webmaster, is recovering well and has updated the FIG UK web site at <http://www.fig-uk.org> and very good it looks too! Not only that, she has found time to write about State Machines in this issue.

I had hoped to be able to bring you six issues of Forthwrite this year, but because of various problems it will have to be just five and so the deadline for the next issue has been put back. If you have any material for publication please send it to me by Wednesday 9th June.

Don't forget the monthly IRC session. Our next ones are Saturday 5th June and Saturday 3rd July on the IRC server called "IRCNet", channel #FIGUK from 9:00pm BST (that's 2000 UTC for international participants, see: <http://www.timeanddate.com/worldclock/>).

Until next time, sally Forth,

Graeme Dunbar



Forth News

Graeme Dunbar

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

Forth Events

euroFORTH 2004

The dates for this year's euroFORTH conference have not been confirmed yet, but the likely dates are the 19th to the 22nd of November 2004, to take place at Castle Dagstuhl in Saarland, Germany. For anyone considering taking part there is a mailing list for delegates at:

<http://groups.yahoo.com/group/euroforth/> The list has been very quiet of late, with (at the time of writing) the last posting being in early February.

FIG UK 25th Anniversary Reunion

FIG UK will be holding its Silver Jubilee this November. Details to follow.

Forth Resources

ACM SIGPLAN Notices

With print material on Forth topics becoming increasingly hard to find it is refreshing to come across recently published work on the net. Paul Frenger has two articles: "Forth report: Deus Ex Macintosh" and "Forth: Dutch

treat" published this year in SIGPLAN Notices. They can be found on-line through their portal at: <http://portal.acm.org/portal.cfm> by searching their archive for "forth"

FIG UK Forth CD

Douglas Neale writes: "I am happy to report that the FIG UK CD is nearing completion and only awaits the final selection of the Chuck Moore Lecture videos with which to fill the space available. The probable price of the CD will be £6."

"The current software and archive space on the CD comes to about 200MB or so, which means we have nearly 500MB free for the videos (I've bought a job lot of 80 minute blanks)."

Jeremy Fowell reports that he has downloaded the list of Forth related videos on Jeff Fox's web site

(<http://www.ultratechnology.com/rmvideo.htm>)

"There are 11 by Chuck Moore and quite a few more by other people."

Douglas adds "I haven't had very much input from members, which is a shame."

No takers for the label design award then!

Situations Vacant

Forthwrite plays a central role in communicating ideas within FIG UK. To make sure that it continues to do so, new blood is needed to help with its production. The editor is looking for FIG UK members to take on authorship of some of the regular columns.

'Forth News' Reporter

The Forth News column reports on News and Current Affairs in the Forth Community worldwide. The format of the column is flexible and although it is desirable to retain the current "edited headlines" to inform the readership of events, there is also scope for more in-depth reviews and comment on current affairs in Forth circles.

'From the Net' Correspondent

Forums like UseNet's comp.lang.forth reflect the current interests of Forth programmers. Forthwrite requires someone to "lurk" (or take an active part) on c.l.f. and other hot spots and report on recent activity. The pieces in Forthwrite to date have tended to summarize a selection of interesting threads. The correspondent could follow this or any other suitable alternative approach to keep FIG UK members up to date with current hot topics.

The columns need not be long and the task should not be an onerous one. The means of submitting copy to the Editor is flexible. It could range from plain text in emails to edited files.

If you would like to join the regular contributors to Forthwrite and fill one or other of these posts, or if you have an idea for a new regular column that you would like to contribute, then please contact the Editor.

The Editor

Membership Matters

Douglas Neal & Graeme Dunbar

The Secretary and Editor take a look through the recent membership applications and renewal forms and report what members are saying.

We welcome new member Iain Shearer (M2068), who, coincidentally "stays" (as we say in this part of the country) not far from the Editor. Please write in to let us know what your interests are when you get the chance Iain.

John Matthews (M120) rejoined just recently. John is author of the book "Forth: applications in engineering and industry", now sadly out of print like so many good Forth books. He has been busy in other fields that he hopes to link with Forth when the time comes.

The member representing Micross Electronics Ltd (<http://www.micross.co.uk/> and <http://www.laundryautomation.com>) is now Nicholas Nelson.

John Craston (M1257) writes "After nearly twenty years away from Forth I have just ordered Forth from MPE - starting again at 81 - I still think it's wonderful."

Ray Allwright (M301) adds: "I have installed Linux (Debian 3.0) on my PC. Struggling a bit to get it properly set up. I am still relying heavily on DOS and a little on MS Windows, but hope to declare a Windows-free zone one day." Can anyone offer help?

Interestingly, Jeff Penn (M2027) lists Debian as one of his interests, along with colorForth, eForth FreeBSD and F11-UK.

Steven Smith (M194) notes that he uses or has used ANSI Forth, polyFORTH, SwiftX and SwiftForth professionally and privately and Paul Bennett's (M1315) expertise lies in high integrity distributed embedded control systems.

It has been our general policy not to advertise members' addresses or phone numbers, but if anyone wants to contact another member, a message can be passed on through the Editor. Our widespread membership is both a strength and a burden. Unfortunately, being so far flung we have no regular get-togethers where we can meet up. If you want to try to make contact with FIG-UK members in your area then please drop a line to the Editor and it will be posted in this column.

Simple State Machines

Jenny Brien

Jenny presents a means of defining simple ad hoc state machines and then goes on to explore a more sophisticated version based on Julian Noble's Finite State Machines (<http://www.jfar.org/article001.html>).

Some tasks are easy to specify, but hard to program procedurally. Consider the task of accepting numerical input from the keyboard. An unfriendly program lets the user enter the entire number before informing him that he typed two decimal points after the first digit. A friendly program, by contrast, refuses to recognize or display illegal characters. It waits instead for a legal character or carriage return (signifying the end of input). It permits backtracking, allowing erasure of incorrect input.

Example

To keep the example small, our number input routine allows signed decimal numbers without power-of-10 exponents. Decimal points, numerals and leading minus signs are legal, but no other ASCII characters (including spaces) will be recognized. Here are some examples of legal numbers:

0.123, .123, 1.23, -1.23, 123, etc.

The “rules” for converting the numbers can be listed as:

1. The first character can be –, 0-9 or a decimal point. (?SIGN)
2. After the first character, – is illegal. (?WHOLE)
3. After the first 0-9, end-of-input is allowed (?WHOLEEND "-" is not a legal number)
4. After the first decimal point, decimal points are illegal. (?FRACTION)
5. End-of input is only allowed again after 0-9 (?END Just to be awkward, 123. should be 123.0)

If you try writing a single routine to EMIT or DROP a key according to these rules and you will find that there are just too many conditions for comfort. Five in all: a test for each of the three classes of acceptable input, and tests for whether a sign or a decimal point have already been received.

There are five states, corresponding to the three classes. The first tests for all three possibilities, the second for two, and the fourth for only one (0-9). While in the later states, the extra tests need not be performed. Is there a simple way to program that clearly? Yes:

Implementation

Instead of nesting words within each other we can pass control as States on the data stack. In this context a State is simply an execution token, which when executed leaves another State on the stack for each possible exit condition. The program is simply a loop which continuously executes States until one causes a termination by leaving a zero. Thus the general form is:

```
... beginning_state BEGIN DUP WHILE EXECUTE REPEAT DROP
```

In this case a State will be executed on every keystroke:

```
: GETSIGNED          \ display a signed number typed at the keyboard
  ?SIGN BEGIN
    KEY OVER WHILE
    ....           \ handle deletes etc. here
    SWAP EXECUTE REPEAT 2DROP ;

: STATES             \ n ++ ; create n named states
  0 DO VALUE LOOP ;

: RETURN             \ a synonym for "EXIT THEN"
  POSTPONE EXIT POSTPONE THEN ; IMMEDIATE

13 CONSTANT EOL
CHAR . CONSTANT DP
CHAR - CONSTANT MINUS

: 0-9? \ n -- f
  [CHAR] 0 [CHAR] 9 WITHIN ;

5 STATES ?SIGN ?WHOLE ?WHOLEEND ?FRACTION ?END
```

To take the simplest states first:

```
:NONAME \ char -- State ; wait for a digit, then allow end of input
  DUP 0-9? IF EMIT ?END RETURN
  DROP ?FRACTION \ not a digit - try again on next key
;
TO ?FRACTION

:NONAME \ char -- State ; wait for a digit or end of input
  DUP 0-9? IF EMIT ?END RETURN
  eol <> ?END AND \ 0 if key is eol, otherwise stay in same
                  \ state
;
TO ?END

:NONAME \ char -- State ; wait for a digit, then allow end of input
```

```

                                \ or decimal point
DUP 0-9? IF EMIT ?WHOLEEND RETURN
DUP DP = IF EMIT ?FRACTION RETURN
DROP ?WHOLE
;
TO ?WHOLE

:NONAME \ char -- State      ; wait for a digit, eol, or dp
DUP 0-9? IF EMIT ?WHOLEEND RETURN
DUP DP = IF EMIT ?FRACTION RETURN
eol <> ?END AND
;
TO ?WHOLEEND

:NONAME \ char -- State      ; allow -, digit, eol, dp
DUP MINUS = IF EMIT ?WHOLE RETURN
?WHOLE EXECUTE          \ test the other cases on the same key
;
TO ?SIGN

```

Another Way

This is reasonably straightforward, but it is beginning to become verbose. Several states are identical in the tests made and actions taken, differing only in the state to be executed next. We need to be able to define the action to be taken and the transition to be made separately.

When scanning ASCII text it is possible to do all the tests in parallel by using the character as an index to a lookup table:

```

: LOOKUP \ ++ create an Ascii lookup filter
  CREATE 256 0 DO 0 C, LOOP ;

: FILTER \ char lookup -- char'
  + C@ ;

: INSTALL \ char lo-char hi-char lookup -- ; fill this range in lookup
                                \ with char
  DUP >R CHARS + SWAP R> CHARS + \ char &hi &lo
  DO
  DUP I C!
  LOOP DROP ;

LOOKUP NUMERICAL

1 CHAR 0 CHAR 9 NUMERICAL INSTALL
2 minus        NUMERICAL + C!
3 dp.          NUMERICAL + C!
4 eol          NUMERICAL + C!

```

The number returned can be used as an index into a two-dimensional STATETABLE array holding the action to be taken and the index of the state to be executed next. Each Statetable holds the current value of its state index and updates it on each execution.

```

: STATETABLE \ n ++           ; create a state table for n inputs
  CREATE 0 , ,               \ initial state index is 0
DOES> \ input# -- ?         ; execute action for input# in current state
                                \ and set up next

  DUP >R                     \ save start address
    2@                       \ input# #inputs current_state_index
    * +                       \ index of entry for this input# and state
  2* 2+ CELLS R@ +          \ address of action
  DUP >R @ EXECUTE
  R> CELL+ @ EXECUTE \ -- n   ; action to fetch new state index
  R> ! ;                     \ update state

: | \ ++                     ; compile an xt in data space
  ' , ;

```

```

\ Generic State-transition words
\ All State-transition words leave an index on the stack, which the executing
\ STATETABLE will immediately consume

```

```

0 CONSTANT >0
1 CONSTANT >1
2 CONSTANT >2 ... etc

```

```

\ Special word for this case:

```

```

: FINISH \ key -- 0 0       ; R: statetable ; change key to 0 to exit loop
                                \ when statetable exits it is left in initial state
  DROP 0 0 ;

```

```

5 STATETABLE Signed#
\ input:      | other?      | num?          | minus?        | dp?           | eol?
\ state:      -----
( 0 ?sign)    | DROP | >0 | EMIT | >2 | EMIT | >1 | EMIT | >3 | DROP | >0
( 1 ?whole)   | DROP | >1 | EMIT | >2 | DROP | >1 | EMIT | >3 | DROP | >1
( 2 ?wholeend)| DROP | >2 | EMIT | >2 | DROP | >2 | EMIT | >3 | DROP | Finish
( 3 ?fraction)| DROP | >3 | EMIT | >3 | DROP | >3 | DROP | >3 | DROP | >2
( 4 ?end>)   | DROP | >4 | EMIT | >4 | DROP | >4 | DROP | >4 | DROP | Finish

```

```

: Getsigned BEGIN KEY DUP numerical filter Signed# 0= UNTIL ;

```

Jenny Brien has been
messing about with
Forth for nearly
twenty years.

She is a keen cyclist
and local historian and
lives with her mother
in deepest Ulster.



Regions: into unknown Win32Forth

David R Poachin

Windows GUI programming is a topic many Forth programmers shy away from. Dave Poachin gets to grips with Regions^{1,2}, one of many classes of Windows API functions.

In my Win32For folder, I have a sub-folder **NotWin32**, which contains some unsuccessful attempts to do 'Windows' things in Win32For.

One of these failures is an attempt to use the family of procedures concerning **Regions**, I knew the Windows functions required but could not get them to work, they just did not seem to be in the system.

Using the menu Win32/Display/. Procs displays a list of all the API functions currently available and there is nothing remotely like '**CreateRectRgn**' or any other call with the suffix '**Rgn**'. I recalled the time when it was accepted that much free software was a poor incomplete imitation of the commercial products, so alas, into the Not Win32 folder went all my work on Regions.

Then one day in my black despair, idly clicking through the news group comp.lang.forth up comes a reply by Jos. v.d. Ven to a similar problem about missing functions. In summary use 'WinView/Search/Find Text in Files...' or Ctrl+Shift+F and set up the dialogue as follows.

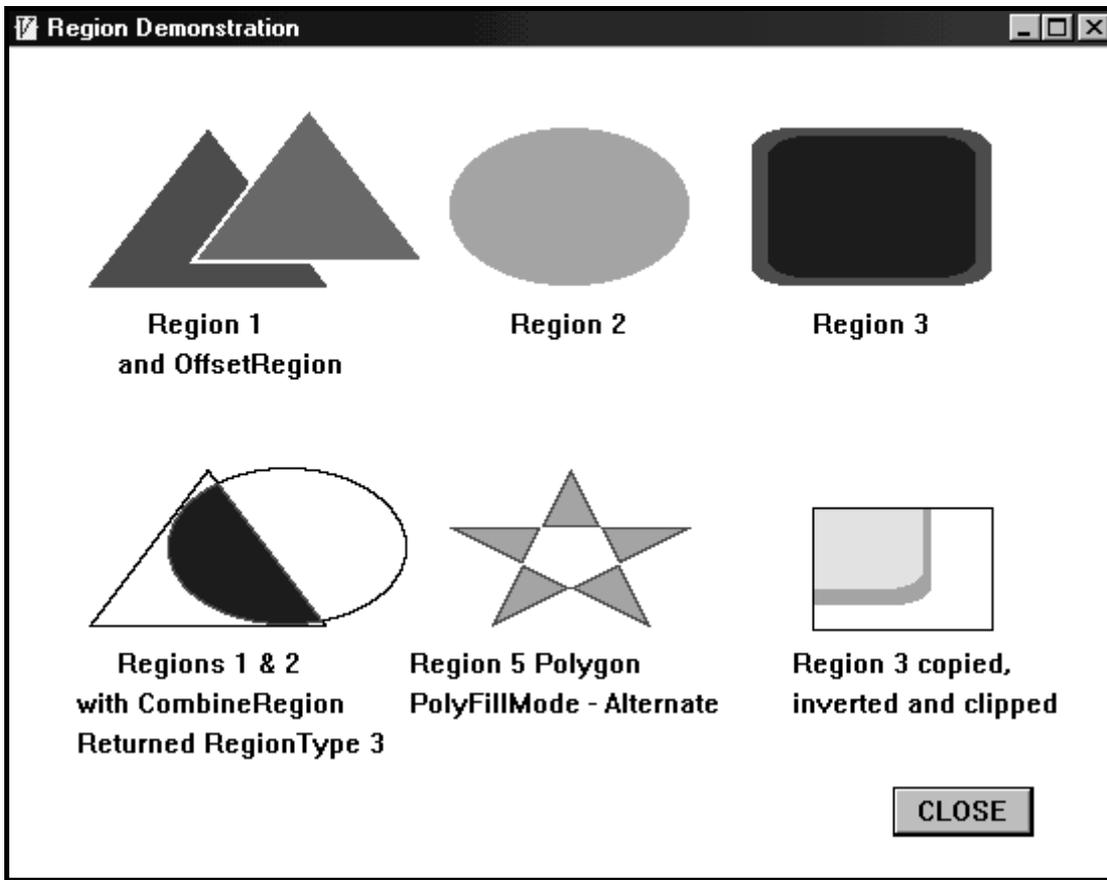
| | |
|------------------------------------|----------------------|
| Text to search for: | 'CreateRoundRectRgn' |
| Files to search: mask, mask | '*.dll ' |
| Directories to search: spec; spec; | ' C:\Windows ' |
| Search all sub directories | checked |

Using Search, returns all the 'dll's' containing CreateRoundRectRgn. One of these is C:\Windows\SYSTEM\COMCTL32.DLL

Returning to Win32 enter '.LIBS ' to view the current libraries, there is no entry for COMCTL32.DLL. Back into deep and black despair again? Not quite, in the file **WinLib.f** there is a word WinLibrary. So enter ' WinLibrary COMCTL32.DLL '. Then using ' .LIBS ' again shows that COMCTL32.DLL is now available, and so all the functions within COMCTL32.DLL should now be accessible.

Many, many thanks to Jos v.d. Ven for the time he spends looking after fellow Forthers. As it happens, **Regions** was not the only topic I could not use. I had always wanted to use the " new " common controls like progress bars, trackbars and the like, but just could not find any way

of getting started, but the main purpose of COMCTL32.DLL is to provide all these common controls. So the Jos v.d.Ven has done me a double service. Further information regarding common controls can be found using the links on the Win32/Help/Forth Web pages/ – Jeff Kelm³ and Michael Hillerstrom⁴, again many thanks.



Back to working with regions...

In the listing, first include a line ' WinLibrary COMCTL32.DLL ' .

Superficially Regions appear to be a variation of Rectangles, but the textbooks say in that the Rectangle functions use an array in data structure and that Regions are GDI objects like bitmaps and brushes, this means that they must be created, that they have handles and that they should be deleted before exit. Win32for does not have any defining words for using Regions, we *have* to use the Windows Call functions and be careful to reverse the order of the parameters, as is usual in Win32For.

Each figure on the screen is formed by a method in the listing, such as M: Region1:... and three types of windows functions predominate.

The Create Calls

There are nine functions for creating regions. The simplest is **CreateRectRgn** that takes the four corners of a rectangle as parameters and returns a handle for the region (or a 0 failure

flag)!. In Win32For the usage is ' y2 x2 y1 x1 call CreateRectRgn '. The listing uses some of the variations of this basic function, using either different or additional parameters.

In Region 1:

CreatePolygonRgn is used and requires three parameters ' PolyFillMode count addr '.

- PolyFillMode takes one of two values, WINDING or ALTERNATE that affects the way a complex Polygon is filled. To show this effect CreatePolygonRgn is used again in Region 6. Change ALTERNATE to WINDING to see the results. You really need a complex figure with many crossing lines and a better technique than mine to achieve any specified result.
- Count is just the number of points in the Polygon.
- Addr is the address of a structure containing the points. Remember to change the Forth address to the absolute address using 'rel>abs'.

In Region 2:

CreateEllipticRgn is used with the parameters of the bounding rectangle.

In Region 3:

CreateRoundRectRgn is used with additional parameters for the curvature of the corners.

The Display Functions

To use a region on the screen requires that the region be associated with the device context of the screen. There are many functions that will do this, all need handles, at least the handle of the Region, and the handle of the Device Context, some require additional handles and parameters. In the listing all is in Win32For order , (i.e., parameters reversed) and the return values have been dropped.

The basic function has the pattern, ' region handle DC handle function ' in a form similar to ' hRgn1 GetHandle: DC Function '.

In Region 1:

In Region1 and later, there are calls such as:

- hRgn1 GetHandle: dc Call **PaintRgn**, which requires two parameters, the handles of the region and of the dc, and uses the current brush.
- hBrush1 hRgn1 GetHandle: dc Call **FillRgn**, which in addition requires a brush handle.
- 2 1 BlackBrush hRgn1 GetHandle: dc Call **FrameRgn**, which requires two additional parameters for the height and width of the frame.

In Region 6:

There are two functions that are easy to write but have complex actions.

- hRgnClip GetHandle: dc Call **SelectClipRgn**, all further display operations are clipped to the region selected by this function.
- hRgn3 GetHandle: dc Call **InvertRgn**, which inverts the colours in the region.

Inter Region Functions

There are a number of functions used to combine, move, or examine regions. These functions only require the handles of the regions involved (up to 3) depending on the action, and maybe extra parameters.

In Region 1:

A typical example is ' -12 50 hRgn1 Call **OffsetRgn** ', where the additional parameters are the vertical and horizontal offset values to move the region.

In Region 4:

The triangle and the ellipse on the top line are moved by offsetting them to new positions. A new region, hRgnA is created, it is rather strange that this region has no size, but it must exist and have a handle to take the result of a combination of other regions. There are five functions that combine regions, four of these take the form:

'combine mode source region 1 source region 2 destination region call combine region',

as in the line:

' RGN_AND hRgn1 hRgn2 hRgnA Call **CombineRgn** to nRgnType '

The CombineMode used here is RGN_AND, but try any of RGN_DIFF, RGN_OR, RGN_XOR to see the variations. In this case, instead of dropping the return value as usual, it is stored as nRgnType, which can take four values indicating the result of combination as follows:

- 0 error, no new region created.
- 1 null (empty) region.
- 2 a simple region, the new region has no overlapping boundaries, it is a rectangle, an ellipse or a simple polygon.
- 3 a complex region is created.

The fifth CombineMode is RGN_COPY, which takes the slightly different form:

'combine mode 0 source region destination region call combine region'

as in:

```
'RGN_COPY 0 hRgn3 hRgnA Call CombineRgn drop'
```

In Region 5

Here is a simple use of **CreatePolygonRgn** using the PolyFillMode ALTERNATE, replace this with WINDING, to see the difference. There is also a more complex function **CreatePolyPolygonRgn** which I may be tempted to use after a little more investigation.

There are many other Windows functions that manipulate regions, including tests to determine whether a region contains a particular point, or if two regions overlap, so there is a great deal more to learn. I think I've got the feel of the topic, and perhaps the most promising and useful feature of regions is the capability to form clipping regions of any shape. In the meantime I must rename that ' NotWin32 ' file.

References

1. Paul Watt. The Code Project: Guide to WIN32 Regions.
<http://www.codeproject.com/gdi/rgnguide.asp>
2. Paul Kuliniewicz. Windows API Reference: Functions.
<http://www.mangovision.com/vbapi/ref/funcc.html#regions>
3. Jeff Kelm. Win32 Programming Examples using Win32For
<http://www.concentric.net/~jkelm/win32for/index.htm>
4. Michael Hillerstrom. <http://users.cybercity.dk/~ccc27382>

Code Listing

```
\ Regions.F      Experiments with regions

anew program
WinLibrary COMCTL32.DLL

:OBJECT Rgndemo <SUPER WINDOW

  ButtonControl Button_1    \ a button

\ Set Up Variables for Brush and Region handles.
\ and for return values

  int hBrush1
  int hBrush2
  int hBrush3

  int hRgn1
  int hRgn2
  int hRgn3
  int hRgn4
  int hRgnA
```

```

int hRgnClip

int nRgnType

\ Set up an Array of Data Points for use with Polyregions
Create POLYDATA
( x1 , y1 , x2 , y2 etc )
40 , 120 , 100 , 40 , 160 , 120 ,

Create POLYDATA2
( x1 , y1 , x2 , y2 etc )
220 , 240 , 340 , 240 , 240 , 290 , 280 , 210 , 320 , 290 , 220 , 240 ,

:M ClassInit: ( -- )
ClassInit: super
;M

:M ExWindowStyle: ( -- style )
ExWindowStyle: SUPER
;M

:M WindowStyle: ( -- style )
WindowStyle: SUPER
WS_BORDER OR
WS_OVERLAPPED OR
;M

:M WindowTitle: ( -- title )
z" Region Demonstration "
;M

:M StartSize: ( -- width height )
550 420 ;M

:M StartPos: ( -- x y )
100 100
;M

:M Close: ( -- )

hBrush1 DeleteObject: dc
hBrush2 DeleteObject: dc
hBrush3 DeleteObject: dc
hRgn1 DeleteObject: dc
hRgn2 DeleteObject: dc
hRgn3 DeleteObject: dc
hRgn4 DeleteObject: dc
hRgnA DeleteObject: dc
hRgnClip DeleteObject: dc
Close: SUPER
;M

:M On_Init: ( -- )

IDOK SetID: Button_1
self Start: Button_1
440 370 70 25 Move: Button_1
s" CLOSE" SetText: Button_1
GetStyle: Button_1
BS_DEFPUSHBUTTON OR
SetStyle: Button_1

\ Create brushes and assign colours

```

```

                255  0  0 rgb call CreateSolidBrush to hBrush1
                0 255  0 rgb call CreateSolidBrush to hBrush2
                0  0 255 rgb call CreateSolidBrush to hBrush3
;M

\ Set up five demonstrations

:M Region1:
WINDING 3 POLYDATA rel>abs Call CreatePolygonRgn to hRgn1
hBrush1 hRgn1 GetHandle: dc Call FillRgn drop
( yinc xinc region handle -- flg )
-12 50 hRgn1 Call OffsetRgn drop
LTMAGENTA BrushColor: dc
hRgn1 GetHandle: dc Call PaintRgn drop
( h w brush handle region handle dc handle -- flg )
2 1 White_Brush GetStockObject: dc
hRgn1 GetHandle: dc Call FrameRgn drop

70 130 s" Region 1" TextOut: dc
55 150 s" and OffsetRegion" TextOut: dc
;M

:M Region2:
( y2 x2 y1 x1 .. )
120 340 40 220 Call CreateEllipticRgn to hRgn2
hBrush2 hRgn2 GetHandle: dc Call FillRgn drop

250 130 s" Region 2" TextOut: dc
;M

:M Region3:
( h w y2 x2 y1 x1 ... )
20 40 120 490 40 370 Call CreateRoundRectRgn to hRgn3
hBrush3 hRgn3 GetHandle: dc Call FillRgn drop
4 8 hBrush1 hRgn3 GetHandle: dc Call FrameRgn drop

400 130 s" Region 3" TextOut: dc
;M

:M Region4:
182 -50 hRgn1 call OffsetRgn drop
1 1 Black_Brush GetStockObject: dc hRgn1 GetHandle: dc Call FrameRgn drop
170 -140 hRgn2 call OffsetRgn drop
1 1 Black_Brush GetStockObject: dc hRgn2 GetHandle: dc Call FrameRgn drop
0 0 0 0 Call CreateRectRgn to hRgnA
RGN_AND hRgn1 hRgn2 hRgnA call CombineRgn to nRgnType
hBrush3 hRgnA GetHandle: dc Call FillRgn drop
1 1 hBrush1 hRgnA GetHandle: dc Call FrameRgn drop

RGN_COPY 0 hRgn3 hRgnA Call CombineRgn drop

55 300 s" Regions 1 & 2" TextOut: dc
35 320 s" with CombineRegion" TextOut: dc
35 340 s" Returned RegionType" TextOut: dc
180 340 nRgnType (.) TextOut: dc
;M

:M Region5:

200 300 s" Region 5 Polygon" TextOut: dc
200 320 s" PolyFillMode - Alternate" TextOut: dc

```

```

    ALTERNATE 6 POLYDATA2 rel>abs Call CreatePolygonRgn to hRgn4
    hBrush2 hRgn4 GetHandle: dc Call FillRgn drop
    1 1 hBrush1 hRgn4 GetHandle: dc Call FrameRgn drop
;M

:M Region6:

    390 300 s" Region 3 copied," TextOut: dc
    390 320 s" inverted and clipped" TextOut: dc

    ( y2 x2 y1 x1 ... )
    292 490 230 400 Call CreateRectRgn to hRgnClip
    hRgnClip GetHandle: dc Call SelectClipRgn drop
    160 -30 hRgn3 Call OffsetRgn drop
    hBrush3 hRgn3 GetHandle: dc Call FillRgn drop
    8 4 hBrush1 hRgn3 GetHandle: dc Call FrameRgn drop
    hRgn3 GetHandle: dc Call InvertRgn drop
    1 1 Black_Brush GetStockObject: dc hRgnClip GetHandle: dc Call FrameRgn
    drop
;M

:M MakeRegions:

    Region1: self Region2: self Region3: self
    Region4: self Region5: self Region6: self
;M

:M On_Paint: ( -- ) \ screen redraw procedure

    MakeRegions: self
;M

:M WM_COMMAND ( hwnd msg wparam lparam -- res )
    OVER LOWORD ( Id )
    CASE
        IDOK OF
            Close: self
        ENDOF
    ENDCASE
;M

;OBJECT

: DEMO ( -- )
    Start: Rgndemo
;
cr cr .( Type DEMO to run )

\ END OF LISTING

```

Letters

Letters to the Editor are always welcome. Please send them the old-fashioned way or by email – which ever is most convenient.

First we have a Win32Forth programming tip from Jim Boyd:

Dear Graeme,

I've noticed that when using the menu to load a file in Win32For that files with spaces in the name or file path will not load. I found the source for the word which is executed by that menu choice and part of it is as follows:

```
s" FLOAD " "pushkeys
```

If FLOAD is redefined like this:

```
: FLOAD      1 WORD COUNT INCLUDED ;
```

then the menu option to load a Forth file can load a file from anywhere on my hard drive even with a filename or path with spaces in it. I do not know if anyone else has discovered this but I hope it helps other users of Win32For.

The best I can do if I need to edit a file with spaces is to open the file after opening WinEd.

I checked this with versions 5.2 Build 0836 and 6.08.00 Build 18.

Regards,
Jim

Thank you and congratulations on winning your FIG UK award, Jim.

Jenny, our Webmaster emailed to say she had spotted an interesting posting on c.l.f on the theme "what would you put in a new edition of *Dr Dobb's Toolbox of Forth?* from Julian Noble". Duly contacted, this is his reply:

Dear Graeme,

Marcel Hendrix proposed the TOC. But I have subsequently pointed out that interesting material has appeared in Forth dimensions, Forthwrite, and also the ACM SIGForth Journal. Also ACM SIGPLAN Monthly Notices has the occasional Forth article, as well as Paul Fraenger's Forth Column. I have even written a guest column myself.

--

Julian V. Noble

Professor Emeritus of Physics jvn@lessspamformother.virginia.edu

^^^^^^^^^^^^^^^^^^^^^^^^^^

<http://galileo.phys.virginia.edu/~jvn/>

Here is the “fantasy league” Dr Dobb’s Toolbox of Forth Table of Contents, based on the text of the original one, updated to 2004. Thanks to Marcel and all the other posters on c.l.f. that contributed to the discussion.

CONTENTS

Editor's Preface ... vii

PART I FORTH-THE LANGUAGE

- 1 The Forth Philosophy ... 3
by Jonah Thomas
- 2 Teaching Forth as a First Language 11
by Michael Coughlin
- 3 ANS Forth and Wordlists ... 13
by Andrew Haley

PART II FORTH PROGRAMS

- 4 WHIST in Forth ... 19
by Marcel Hendrix
- 5 Elements of Forth XML Data-Base Design ... 29
by Elizabeth Rather
- 6 The Forth Sort ... 41
by Brad Eckert
- 7 SENDIT and RECV ... 47
by Stephen Pelc
- 8 Interfaces for a Mouse ... 57
by Alex McDonald and John Passaniti
- 9 Relocating Loader in Forth ... 63
by Bernd Paysan
- 10 Forth Decompiler ... 71
by Albert van der Horst
- 11 Screen-Oriented Editor Re-Visited . 75
by Wil Baden and Leo Brodie
- 12 Evolution of a USB device driver 85
by Bill Gates, PI
- 13 OOFing Screen Editor ... 91
by Doug Hoffman
- 14 The Conference Tree ... 101
by Ward McFarland

PART III MATHEMATICS IN FORTH

- 15 Series Compression in Forth ... 107
by Julian V. Noble, Professor Emeritus
of Physics
- 16 Forth SSE3 Floating-Point Package 113
by Neil Bawd
- 17 Signed Integer Division ... 137
by Robert L. Smith

PART IV MODIFICATIONS/EXTENSIONS

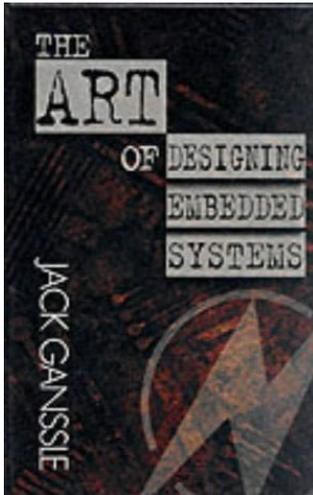
- 18 A Proposal for Strings in Forth ... 143
by Gary Chanson
- 19 Non-Deterministic Control Words .. 151
by Paul Kleinrubatscher
- 20 Some Forth Coding Standards ... 159
by Paul Bennett
- 21 Towards a More Writable Forth Syntax
... 165
by M. Anton Ertl

PART V IMPLEMENTING FORTH

- 22 Forth on hyper-threading CPU architectures ... 181
by Joe Knapka
- 23 A TMS320C40 Forth Assembler ... 189
by Michael L. Gassanenko
- 24 Forth Assembler for the Rabbit ... 199
by William F. Ragsdale
- 25 MARC4 4-bit Forth ... 211
by Willem Ouwerkerk

PART VI APPENDICES

- A Converting Forth-83 Programs for ANS Forth ... 235
by John Hayes
- B The Authors ... 243
- C For More About Forth ... 245



Book Review

"The Art of Designing Embedded Systems"

by Jack Ganssle.

Reviewer: Paul E. Bennett

To those of us who have been hanging around the comp.arch.embedded newsgroup for a long while (at least 10 to 15 years - so long I am not certain) you will have come across some of Jack's postings. You may also have subscribed to his occasional newsletter "The Embedded Muse". Having done both, for as long as I can remember clearly, I have come to treat Jack as a friend despite never having met him and shook his hand. I was therefore delighted to read his book.

When I got hold of my copy I read the introduction then skipped straight to Appendix B (A Drawing System) as I was interested in what treatment Jack was giving to document/information management. The drawing system analogy works well for software no matter how strange it might seem to those who have in the past just hacked out their code without thought for how to store and control the code development. If you do nothing else, implement a document management system and commit your code to it along with any descriptive documents that you have as well. This will eliminate a major problem faced by many developers (being unsure of what their current sources are). A VCS or RCS will also help you in this respect if you feel you need the help of your computer.

The rest of the book deals with disciplined development, the need to limit code size, real time issues, firmware issues, hardware, troubleshooting and tools, and human factors. All offer a common sense approach to the development of embedded systems. Within the introduction Jack admits he is trying to revive the respect for engineering that pervaded the Brunel era. He reminds us, though, that it really is in our own hands to improve our image.

While no Forth is involved in the book (there is not much mention of anything more than the merest hint at C, C++ and assembler) it should grace the Forthists shelves for the useful reminders it can provide.

Title: The Art of Designing Embedded Systems (EDN Series for Design Engineers)
Author: Jack Ganssle
Publisher: Newnes, 1999
ISBN: 0-7506-9869-1
Hardback: 352 pages
Price: £36.99

Start Simple

Graham Telfer

Novices need not feel left out in the cold now with Graham's piece "Start Simple". He will develop the theme in future articles.

Converting a Temperature

Imagine you are going on holiday to America. You want to know the kind of weather you can expect. Being a modern European you think in Centigrade. It comes as a bit of a shock to discover Americans are still using Fahrenheit.

An ideal chance then, to write a Forth program that converts Centigrade to Fahrenheit. This article develops a program to do just that.

Forth has a number of idiosyncrasies all its own. It encourages factoring of procedures (called words by Forth programmers) to a degree beyond what may be considered usual in other languages. As a consequence Forth can be difficult to read if the factoring is done badly.

Another feature of Forth is post-fix notation for arithmetic. This might feel strange to use at first, but is no stranger than the pre-fix notation used by Lisp or Scheme. There is a good reason for using post-fix; Forth uses a data stack to pass parameters and the parameters must be present on the data stack before the arithmetic function can be applied. For example:

```
5      goes on the stack
4      goes on the stack
+      pops them off and puts the result back on the stack
9
```

This looks like the way you did adding in school.

Forth likes Integers

In classic Forth style this first program will use integer arithmetic. First of all we need to know how to convert between Centigrade and Fahrenheit. If you look the formula up in a text book you probably found this:

$$\text{Centigrade} = (\text{Fahrenheit} - 32) / 1.8$$

or

$$\text{Fahrenheit} = (\text{Centigrade} * 1.8) + 32$$

The problem with this is the 1.8. Yes we can scale the decimal point away; correcting later, but there

is a formula that uses just integer arithmetic that will do the job for us without scaling:

$$\begin{aligned}\text{Centigrade} &= (\text{Fahrenheit} - 32) * 5 / 9 \\ \text{Fahrenheit} &= (\text{Centigrade} * 9) / 5 + 32\end{aligned}$$

Remember that this program is going to be used by a person going on holiday. Fractions of a temperature are not important.

A Short Specification

Even though this program is simple, writing a short specification is always worth doing for future reference.

There are two clear words needed in this application. One will take the temperature and convert it from Centigrade into Fahrenheit, and a second word to display the result.

I have decided to include a constant in the specification. That is the freezing point of water. In Fahrenheit it is 32 degrees.

$$\begin{aligned}\text{Centigrade} &= (\text{Fahrenheit} - \text{FreezingPoint}) * 5 / 9 \\ \text{Fahrenheit} &= (\text{Centigrade} * 9) / 5 + \text{FreezingPoint}\end{aligned}$$

FreezingPoint: Integer = 32

Fahrenheit->Centigrade: Integer -- Integer

Centigrade->Fahrenheit: Integer -- Integer

Display: Integer --

The Code

Notice how a Forth word begins with a colon and ends with a semi colon. These are Forth words just like any other. Forth words are always separated by at least one space.

```
32 constant FreezingPoint
: Display ( result-- ) . ;
: To->Fahrenheit ( centigrade-- )
  * 9 5 / FreezingPoint + Display ;
```

To use this program simply enter the Centigrade temperature followed by the Forth word and press the Enter key:

```
24 To->Fahrenheit
75 ok
```

The result is displayed as a single integer by the tiny word:

```
Display
```

The name of the program works well with how we speak. Convert 24 to Fahrenheit please.

Finding suitable names is an art by itself. We will make one modification to the program. Forth has an operator `*/` which can replace the `* 9 5 /` in the code. The benefit of making this change is that any overflow caused by the multiplication by 9 before the division by 5 is avoided. Forth keeps the intermediate result on the data stack as a double number. As an extra touch we can display the result on a line lower by adding the word `CR` before the word `Display`.

In the early days of Forth, 8 bit computers were common and so this feature was important. It might be argued that today, with 32 or 64 bit computers the combined operator is redundant. It is still useful to know about however. The code with this change made is:

```
32 constant FreezingPoint
: Display ( result -- ) . ;
: To->Fahrenheit (centigrade -- )
  */ 9 5 FreezingPoint + CR Display ;
```

Improvements

There are no checks on data in the code as it stands. A user could easily try entering a letter instead of a number. Neither have we tested the program. Are we sure that it will handle sub zero temperatures correctly for example?

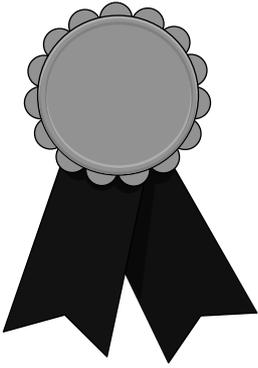
The user interface is pretty much non-existent. Developing a user interface for this article is the subject of the next article. In the meantime you might try writing the other half of the conversion; Fahrenheit into Centigrade for all those poor Americans coming on holiday to Europe.



I learned Forth in the days of the Amstrad PCW. I can remember how thrilled I was to get a stick figure to walk across the monitor. I have learned a few other programming languages along the way, (Prolog being a favourite) but Forth remains one of the languages I enjoy using. I write programs for fun, because some problem interests me and I want to explore some aspect of it. Forth is good for that.

I am sorry that Forth has become so closely identified with embedded systems. Somewhere along the line a great opportunity was lost to make Forth the general purpose programming language of choice for amateurs and professionals alike.

Being able to program your PC liberates you from the clutches of the Microsofts of this world. Programming is a core skill I believe everybody should learn. Forth remains one of the languages that is accessible. The idea of *programming with Words* is a powerful one, that needs more emphasis in teaching Forth.



Presenting the FIG UK Awards of 2003

These awards are given to encourage effort recognise achievement.

The FIG UK Awards of 2002 were won by Ed Hersom and Howerd Oakford.

Free
membership

To everyone who sent in their nominations – “thank you”. Looking back, a lot of good work was done during 2003 and our judges, the officers of FIG UK, have now chosen two winners. They each receive:

- a place in the FIG UK web-site's Hall Of Fame
- this mention in Forthwrite
- ***a year's free membership.***

Achievement

Henry Vinerts: his entertaining and enthusiastic epistles from across the pond on the activities of the Silicon Valley Forth Interest Group are always a delight to read.

Forthwrite

James Boyd: a newcomer to FIG UK, for his two part article entitled “A Virtual Nondeterministic Machine in Forth” showing us how Forth can go so far outside its normally recognised boundaries.

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).

Dear friends,

It bothers me that two months have passed without my Forth signals to you from Silicon Valley, and now I have missed the submission deadlines for both the Vierte Dimension and Forthwrite. I am sorry about that, but I'll make the excuse that our usual schedule of SVFIG meetings on the fourth Saturday of the month is at least partly to blame for the delay. (*Henry's contribution arrived in plenty of time. It is Forthwrite that is behind schedule – Editor.*)

Silicon Valley FIG Meeting – February 2004

For me, the highlight of our February meeting was Bob Smith's lecture on musical scales and various systems of tuning ("temperament") of keyboard instruments. He has written a Win32Forth program that calculates the deviations from given temperaments in "cents." (The chromatic scale of a twelve-tone octave equals 1200 cents, a cent being equal to the musical interval of one-hundredth of a half-tone.) Using sample sound files that play musical thirds in various temperaments, Bob demonstrated to us the effects of tuning to different schemes. As I have mentioned before, Bob is not only a major contributor to FPC and Win32Forth, but also an accomplished accordionist, a man with a lot of knowledge he willingly shares with others.

Dr. Ting presented a brief explanation of a clever algorithm he has developed for drawing ellipses, in Forth, of course, and David Frech returned to more fully describe his muForth system. Again, good educational lectures, albeit generally above the level of easy comprehension for yours truly--the oldest Forth novice in the world. I must tell you that by now I have almost overcome the uncomfortable feeling of being assumed Forth- and computer-literate like the rest of the group. I merely apologize to friendly individuals when they try to converse with me in a language that I don't understand. After David's talk, one helpful newcomer didn't sense that his thorough explanation of "tail-recursion" was going to the wrong ears. I hope that he comes back, so I can thank him again.

OK, that brings me to last Saturday, the March meeting.

Silicon Valley FIG Meeting – March 2004

First of all, I have to record my surprise of seeing four new faces among the twelve people who were there at 10 a.m. We have had new visitors lately and a noticeable drop in regular attendance, but the ratio of new versus old was certainly unusual today.

Dr. Haydon asked me to note in my letter that all of the remaining FIG materials and documents, some 78 shipping cartons in all, have now been transferred to Mountain View Press and are available for purchase. An inventory is being prepared and should appear soon on the Web. (In the meantime, you can send inquiries to Glen directly after finding him at TheForthSource.com.)

Dr. Ting started the day with a description of a single-step debugger for the eP32 Forth simulator. This is for debugging the hardware code on the chip which is designed in Taiwan, and works on the principle of single-stepping Forth words rather than machine instructions. "In software, Forth is its own debugger," Ting says.

A few more people showed up and participated in various unscheduled discussions and Web visits, but only about a dozen returned after a long lunch to listen to John Hall, the former FIG president, who has been working for Apple for a number of years now, and came to report the good news that there is more work for Forth in the ever-growing Open Firmware applications that exist in all Macintosh machines. As a matter of fact, Apple has even sponsored a Forth course for their programmers. For my readers, I want to report that GForth is the primary Forth being used at Apple. I also want to take this opportunity to question whether any other forther can match John's experience of having worked since 1980 only for companies where he was allowed to do Forth. If he wasn't, John says, he would leave and pick another company that allowed him to.

I don't have enough space, nor did we have enough time left in the day, to describe the accomplishments of the last speaker, Sellam Ismail. This energetic young man is the owner of Vintage Tech and the organizer of Vintage Computer Festivals, that have been held in this country since 1997. I understood that the European version of this festival will be this May 1st and 2nd in Munich, and I shall venture a guess that Sellam's friend--his European counterpart--might be Hans Franke of the "Gesellschaft fuer historische Rechenanlagen," whom I have met on two occasions in Silicon Valley's VCFs.

Sellam started his computer collection and education with a Mattel Aquarius, 1983, 4K model, running BASIC. By now he has collected about 2000 computers, 900 of which are unique, 7000 computer books, and some 20,000 computer magazine issues, and his storage area has grown from one bedroom to a large warehouse in Livermore, CA. He has

interests and connections in computer recycling and refurbishing businesses, in educational classes for children, and even (reluctantly) in renting vintage machines to the movie industry. It seems that it was our Dave Jaffe's donation of a large supply of vintage computer materials to Sellam's venture that connected us with him. Look for VCF Nr. 7 coming this year again at the Computer History Museum in Mountain View, California.

Whether you print this or not, I wish you all a Happy Easter, and to my German friends, a successful and pleasant conference on the island of Fehmarn.

Henry

(Of course we will print it Henry! It is always good to hear from you and it is fascinating to learn of the wide variety of interests in SVFIG. I for one wish I had been at the demonstration of the microtonal musical scales – Editor.)

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

Library Notes

Graeme Dunbar

News and Comment from the FIG UK Librarian.

Loans

The library is still in a bit of a muddle after the loss of the card catalogue during the recent move. However, it is “business as usual” and loans will be sent out as soon as possible. If you have any enquiries about borrowing books or back issues of Forthwrite please drop me a line.

Catalogue

It is hoped that work will start on rebuilding the catalogue this summer. It is hoped that it will be in MS Access, though Excel may well suffice for our needs. However I would be very interested to hear from anyone with experience of client or

server side databases to look into the feasibility of an on-line catalogue.

Donation

Retired member, Eric Morris, has kindly donated his collection of Forth books and back issues of Forthwrite to FIG UK. The material is currently with Chairman, Jeremy Fowell, who is documenting the collection. The Library may be in the position to dispose of past issues of Forthwrite to interested members when we have established what duplicates we have.

Thank you Eric, for your kind donation.



Vierte Dimension 4/2003

Joe Anderson

Joe provides a look at the latest issue of the German FIG magazine.

Editorial.

4

Friederich Prinz

Friederich Prinz requests articles for Vierte Dimension and draws attention to the Jubilee Year 2004 with proposed celebration "Forth Event 2004" on Fehmarn.

Readers' Letters.

5

Four Letters to the Editor from: Ulrich Paul (HOLON lives), Ulrich Paul again, (compile or interpret), Ulrich Paul again (Forth and Java), Ulrich Paul again (another name for the SWAP-dragon).).

Signs of life.

7

Henry Vinerts

[\[Volvoid@aol.com\]](mailto:Volvoid@aol.com)

Henry reports on the September meeting of SVFIG. Dr. Ting was missing. Surprise guest was Bernd Paysan from "good old Germany", who reported to those present on developments in GForth.

Reviews.

8

Fred Behringer

[\[direktorium@forth-ev.de\]](mailto:direktorium@forth-ev.de)

Fred reports on Forthwrite 122 and Vijgeblaadje 40.

Forth, USB and a Web-server on a Smartcard.

10

Bernd Paysan

[\[direktorium@forth-ev.de\]](mailto:direktorium@forth-ev.de)

Smartcards don't use (up to now) standard PC serial interfaces. They use card-readers. The author reports on an innovation: a Smartcard from Siemens with a USB-connection, a 8051 Processor, some ROM, EEPROM and RAM. The firm Giesecke and Devrient (forgery-proof printing of banknotes and identity cards) has written a USB-driver for this, that prepares a sort of serial interface. With this driver the firm has taken inspiration from the author's "Web-server in Forth".40.

Four-in-a-line.

11

Bernd Paysan

[\[direktorium@forth-ev.de\]](mailto:direktorium@forth-ev.de)

There are two players involved and a board of 7 columns and 6 rows. The players can let their pegs (of different colour) drop in either one of the 7 columns. Each player, one by one, puts a peg on the board. Winning strategy is four pegs of the same colour in a line (left-right, up-down, or diagonal in one of the two possible choices). Each player's opponent can prevent the player's putting "four in a line" by trying to put a peg of his own colour at the end of a threatening line of

three (or less).

This is a game of strategy, which can be played by a computer against a human opponent or another computer. Bernd explains the Alpha-Beta/Min-Max algorithm (a search in a tree with min-max evaluation with the help of a Forth programme). His graphic system MINOS is put to use in the presentation.

Advertisements.

13

Notices advertising for membership of the FIG UK and the Dutch Forth-gebruikersgroep.

A Target Compiler based on a Virtual Machine.

14

Ulrich Paul

There are far too many processors for it to be worth while writing for each a Forth system of its own. Moreover these low-cost processors are installed in systems where every byte counts. The author starts from a host computer with a full set of instructions and turns his attention to the compilation of the target system.

He divides the possible target-systems into levels 0 to 4. The more the target is capable of, the less the host needs to be able to, and vice-versa. The degree of complication of the target-system to be compiled depends on the availability of space in the target. The author suggests a compiler that with few changes (in configuration status) is adapted to all target demands.

Echelons Neuron.

18

Rafael Deliano

The author reports on the American firm Echelon and their "Neuron" controller range. The controllers contain three CPUs, which are identically built, and access memory and I/O over a common bus. The application-processor is programmed by the user. The Media-access-Control processor (MAC) and the network-processor are driven by the protocol software included. The MAC works on the data-handling aspects of the hardware, the network-processor makes the protocol. The three CPUs communicate with each other over the RAM memory.

Fletcher-Checksum.

21

Rafael Deliano

The technique of Fletcher ("An Arithmetic Checksum for Serial Transmission", IEEE Trans. Com., 1982) is discussed. The technique has advantages over the usual CRC calculation. The discussion focuses on the concept of ones' complement. The algorithm is presented in flow-chart form and in 6502 assembler under nanoForth.

Catch and Throw.

24

Filippo Sala

A very readable introduction into the meaning and effectiveness of the pair of ANS-Forth words CATCH and THROW: particularly critical words in a programme that on the occasion of an error that would crash the system would be embedded in a pre-installed CATCH and

subsequent THROW. CATCH in the event of a mistake issues a centrally-controlled error message, THROW latches on to the mistake and makes sure that the programme can still continue after the unsuccessful attempt. All this is in the ANS-Forth specification, but the author gives a comprehensible working-over. It is intended for a second part to follow with possible applications of CATCH and THROW.

Development of USB with Forth.

25

Carsten Strotman

A USB adapter for the Atari XL/XE has been developed. (At the heart of the Atari XL/XE is the 6502 8-bit processor) The drive software is written in FIG-Forth for the 6502. A circuit diagram and illustrations of the adapter are given. The project was suggested by some prior developments of a Dutch member of the Atari Bit Byter User Club, <http://www.abbuc.de>. Details can be found on the author's project page, <http://www.strotman.de/twiki/bin/view/APG/ProjUSBCart>.

Readers' Letters.

29

Another four letters, sent in by: Michael Kalus (the dragon in Fehmarn), Graeme Dunbar (Back to Forth in the class-room), Fred Behringer (introduces Graeme as contact-man for FIG UK), Martin Bitter (advice to Graeme on robot construction, <http://www.stiquito.com>).

Love and Tensor Algebra.

30

Stanislaw Lem

We just have to have our fun: recounting of a very erotic poem by the well-known science-fiction author Lem, in which, though, the main ideas are borrowed from higher mathematics. Impossible? No. On the contrary. Interesting how much more powerfully the imagination is aroused through it.

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



FIG UK Contacts and Information

| | | |
|-------------------------------|------------------------|---|
| 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 (temporary) | Graeme Dunbar, | School of Engineering, The Robert Gordon University, Schoolhill, ABERDEEN AB10 1FR 01224 262415 g.r.a.dunbar@rgu.ac.uk |
| 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, | School of Engineering, The Robert Gordon University, Schoolhill, ABERDEEN AB10 1FR 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 126: 9 June 2004

Issue 127: 18 August 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)