More Forth-41 and HHC2008 Challenge Fun

*Message #1 Posted by **Egan Ford** on 16 Oct 2008, 7:21 p.m.*

I have concluded my Forth-41 testing. Without documentation I think I have reached the end of what I can figure out on my own. Any more work would require a significant time commitment.

Observations:

1. Forth-41 is a bit unstable. Twice I lost most of my dictionary. I think it was related to the RPN SIZE *nnn* command. The best way to start out is to clear all 41 memory, SIZE 010, then XEQ FORTH. FORTH will only use 255 registers and 126 EM registers.
2. Forth-41 is a bit more unstable. While entering the program below words A, B, and C would suddenly vanish. After entering SORT, I checked for words A, B, and C and reentered any missing. This happened twice after completely starting from scratch.
3. Forth-41 needs more memory. I've been unable to utilized more EM. A 1984 DATAFILE article stated that all EM could be used. GROW didn't grow anything. (Did not return error either).
4. Performance. Measurably slower than RPN, see results below.
5. Awkward code entry. I've tried V41, EMU41, and i41CP+. Only i41CP+ has a reasonable Forth-41 input system. This is because it has a QWERTY keyboard AND a touch screen.
6. Many Forth-79 words missing.
7. Can run RPN commands that act on the RPN stack. Interesting, reminds me of Forth for the 71B.
8. No method to backup/restore work (that I know of).
9. Unsure how to use editor.
10. Still a lot of fun. :-)

Thanks to Diego we all have a CX version of this module. This helped out with timing.

The program below is a Forth-41 entry for Gene's programming challenge. It actually works with other Forth systems, but is a bit too large for Forth-41. With some effort I can make it fit, but not today.

Wrapper code to launch and time:

```
01 LBL "FS"
02 RUNSW          ; Start SW
03 FEX            ; Forth Execute what ever is in ALPHA
04 FTOX           ; POP integer from Forth stack in push to RPN stack
05 STOPSW         ; Stop SW
06 END
```

Before running, execute 0 SETSW, when done with all tests, run RCLSW ATIME24 AVIEW. To execute a test, input the points (space delimited), the number of points, and the string FS in the ALPHA register, e.g.:

```
56 66 1 3 FS
```

Then XEQ "FS". **NOTE**: The RPN "FS" and the Forth "FS" names are just a coincident and do not need to match.

Results:

```
i41CP+ Full Speed RPN all 7 problems:   10.88 seconds
i41CP+ Full Speed Forth-41 5 problems:  26.84 seconds
```

Because of space limitations I could not fit in the HEX code. If I was less lazy I could FORGET PG and enter in HX, time and add to my total time, but my point has already been made.

My RPN, Forth, and RPL programs all used the same algorithm. However, the Forth version has a point value limit of 128. The SORT routines used for RPN and Forth are the same (Selection Sort).

NOTE: If you think this looks like my RPL program--it does. I simply cut and pasted it in, then created a number of Forth words to minimize the amount of redundant code.

Code:

```
DECIMAL
VARIABLE N
VARIABLE M
VARIABLE RR
VARIABLE P 6 ALLOT

: A P C@ ;              ( NOTE: May dissapear, check for A-F before coding TR )
: B P 1+ C@ ;
: C P 2 + C@ ;
: D P 3 + C@ ;
: E P 4 + C@ ;
: F P 5 + C@ ;
: R RR @ ;
: UR R DUP 1- 2 */ ;
: DR R DUP 1+ 2 */ ;
: LD R 2 * + 1- 2 */ ;
: CA DUP N ! 0 DO I P + C! LOOP ;
: <= OVER OVER < ROT ROT = OR ;

: SORT
  N @ 1- 0 DO
    I M !
    N @ I 1+ DO
      I P + C@ M @ P + C@ < IF
        I M !
      THEN
    LOOP
    I P + C@ M @ P + C@
    I P + C! M @ P + C!
  LOOP
;

: RW
  1 RR ! 1 M !
  BEGIN DUP M @ > WHILE
    1 RR +! RR @ M +!
  REPEAT
  DROP
;

: TR
  B RW
  C B - B A - > IF
    A UR > IF
      B A - DUP LD B + C = IF
        DROP 1
      THEN
    THEN
  ELSE
    C DR <= IF
      B C B - DUP NEGATE LD - A = IF
        DROP 1
```

```
      THEN
    THEN
  THEN
;

: PG
  C RW B UR > IF
    C B - DUP LD C + D = IF
      B C B - DUP NEGATE LD - A = IF
        DROP 2
      THEN
    THEN
  ELSE
    D C - B A - = IF
      D DR <= IF
        B RW A UR > IF
          B A - DUP LD B + D = IF
            DROP 2
          ELSE
            1 RR +!
            B A - DUP LD B + D = IF
              DROP 2
            THEN
          THEN
        THEN
      THEN
    THEN
  THEN
;

: HX
  F E - B A - = IF
    F E - 2 * D C - = IF
      F RW E UR > IF
        B RW A UR > IF
          B A - DUP LD A + C = IF


( OUT OF MEMORY "BF", cannot use GROW, limited to 255 REGs )


            B A - DUP RR +! DUP LD D + F = IF
              DROP 3
            THEN
          THEN
        THEN
      THEN
    THEN
  THEN
;

: FS
  CA SORT 0
  N @ 3 = IF TR THEN
  N @ 4 = IF PG THEN
  ( N @ 6 = IF HX THEN )
;
```

Edited: 16 Oct 2008, 7:26 p.m.

[Return to Index | Top of Index]

Go back to the main exhibit hall