


```

\ Preset Variables:
0 PWM ! \ not used for now
0 SWI ! \ not used for now
9 PSWI ! \ Set this combined variable to 9 = 1001 this will show the PWM LED ON and T1 ON
0 OUTP ! \ All bits LOW 0000
3 IN ! \ The IN variable set to 3 0011 - prepared to try AND OR XOR INVERT
F ANI ! \ Set to F, so all bits HIGH, free for any use, e.g. program an extra counter, or for other tests
F HOU ! \ not used for now
FFFF ALL ! \ ALL will be the combined 16 Bit Vector PWSI OUT IN ANI later , was commented out here for now

```

```

\ Control Words ( from left to right as shown in LINE2 )

```

```

: PWH PSWI @ $8 OR PSWI ! MBV2 ; \ Get the variable onto stack, set bit 3 using OR, and store it back, now update the Display
: T3H PSWI @ $4 OR PSWI ! MBV2 ; \ Get the variable onto stack, set bit 2, and store it back, see above
: T2H PSWI @ $2 OR PSWI ! MBV2 ; \ And the same for setting all of the other bits
: T1H PSWI @ $1 OR PSWI ! MBV2 ;
: O3H OUTP @ $8 OR OUTP ! MBV2 ; \ x x x x 1 0 0 0
: O2H OUTP @ $4 OR OUTP ! MBV2 ; \ 0 1 0 0
: O1H OUTP @ $2 OR OUTP ! MBV2 ; \ 0 0 1 0
: O0H OUTP @ $1 OR OUTP ! MBV2 ; \ 0 0 0 1
: I3H IN @ $8 OR IN ! MBV2 ;
: I2H IN @ $4 OR IN ! MBV2 ;
: I1H IN @ $2 OR IN ! MBV2 ;
: I0H IN @ $1 OR IN ! MBV2 ;
: A3H ANI @ $8 OR ANI ! MBV2 ;
: A2H ANI @ $4 OR ANI ! MBV2 ;
: A1H ANI @ $2 OR ANI ! MBV2 ;
: A0H ANI @ $1 OR ANI ! MBV2 ; \ here end the words that Set Bits HIGH - the next ones set the same bits LOW using AND
: PWL PSWI @ $7 AND PSWI ! MBV2 ; \ x x x x 0 1 1 1
: T3L PSWI @ $B AND PSWI ! MBV2 ; \ 1 0 1 1
: T2L PSWI @ $D AND PSWI ! MBV2 ; \ 1 1 0 1
: T1L PSWI @ $E AND PSWI ! MBV2 ; \ 1 1 1 0
: O3L OUTP @ $7 AND OUTP ! MBV2 ;
: O2L OUTP @ $B AND OUTP ! MBV2 ;
: O1L OUTP @ $D AND OUTP ! MBV2 ;
: O0L OUTP @ $E AND OUTP ! MBV2 ;
: I3L IN @ $7 AND IN ! MBV2 ;
: I2L IN @ $B AND IN ! MBV2 ;
: I1L IN @ $D AND IN ! MBV2 ;
: I0L IN @ $E AND IN ! MBV2 ;
: A3L ANI @ $7 AND ANI ! MBV2 ;
: A2L ANI @ $B AND ANI ! MBV2 ;
: A1L ANI @ $D AND ANI ! MBV2 ;
: A0L ANI @ $E AND ANI ! MBV2 ;

```

```

\ Set I1 and/or I0 of the INPUTs, then call AND01, OR01, XOR01, INVERT0 and see the result of the logic result in OUT0

```

```

: AND01 IN @ DUP 1 RSHIFT AND 01 AND OUTP ! MBV2 ; \ Do an AND of IN bit0 and bit1, 00=>1 01=>0 10=>0 11=>1
: OR01 IN @ DUP 1 RSHIFT OR 01 AND OUTP ! MBV2 ; \ Do an OR of IN bit0 and bit1, 00=>0 01=>1 10=>1 11=>1

```

```

: XOR01   IN @   DUP 1 RSHIFT XOR 01 AND  OUTP !  MBV2   ;   \ Do an XOR of IN bit0 and bit1, 00=>0 01=>1 10=>1 11=>0
: INVERT0 IN @   INVERT          01 AND  OUTP !  MBV2   ;   \ Do an INV of IN bit0,          0=>1 1=>0
\ Forth Words used - remember: Any Forth Word consists of a character sequence plus a space (not allowed in Words is Space )
\ 0 INCLUDE INCLUDE loads and starts a file in VFX; set up c:\VFXTESTAPP, store file there, type INCLUDE c:\VFXTESTAPP\VFXTESTAPP.f
\ 1 HEX      Mostly VFX recognizes numbers as decimal or HEX. If in DECIMAL mode any Hex Number will give an error. Careful!
\ 2 \       \ and a Space after it tells Forth, that the rest of the line is for documentation, so is ignored by VFX.
\ 3 :       : starts new Forth Word definition, is terminated by ; see Number 6. Example : Bell 07 EMIT ; sends a sound.
\ 4 ."      ." and a space starts a string of characters to be sent to the screen, the string is terminated by "
\ 5 CR      CR defines a word that places the cursor to the beginning of the next line
\ 6 ;       ; ends the definition started by : see number 3
\ 7 TEST    TEST is the name of a new word defined just to display the GUI - the Graphical User Interface
\ 8 Variable Variables = named memory locations. VARIABLE xx defines, nn xx ! sets, xx @ . prints. Remember: NAME and contents
\ 9 DUP     DUP takes the value on top of the stack and puts the same value on top as NEW TOS - Top of Stack
\ 10 $n     $n - the $ ensures that VFX takes the number following as hexadecimal number
\ 11 AND    AND is a logical operator. works on the two top numbers on the stack, on a bitwise basis, only leaves result
\ 12 IF     IF is not zero do A - ELSE do B - THEN continue
\ 13 ELSE
\ 14 THEN
\ 15 LSHIFT LSHIFT does a bitwise shift of the value on stack, expects the number of shifts to be done on stack - 1 LSHIFT
\ 16 SPACE  SPACE sends a space to the screen
\ 17 DROP   DROP does the opposite to DUP; DROP takes the TOS and deletes what was there, all values move up one position
\ 18 DV     DV is a word that combines the 4x 4 bits of the variables for the screen, 16 Bit wide.
\ 19 @     @ stands for me for AT, for example with Variables - xx puts address of xx onto the stack, xx @ gets the value of it
\ 20 ?DO   ?DO will start a DO word word LOOP, expects the number of loops on stack. ?DO checks if number on stack is 0.
\ 21 LOOP  Goes back to ?DO until 0, then continues
\ 22 PAGE  PAGE clears the screen, takes the cursor to the top left position and prints ok
\ 23 BEGIN BEGIN starts a loop of words which ends with UNTIL. UNTIL expects a flag on stack; not 0 then to BEGIN else continue
\ 24 1+    1+ is put on stack, adds one to the number below and replaces it. Same function as 1 + as 2 words.
\ 25 !     ! is the opposite function to @, and stores the number on stack at a location on stack, e.g. 55 xx ! at Variable xx
\ 26 MS    MS stands 1 Millisecond delay in software. Careful which base you are in - 256 and 100 the same Dec / HEX
\ 27 KEY?  Key? Checks, if a key has been pushed as this generates a flag. BEGIN xxxxxxxx KEY? UNTIL - flag stops the loop
\ 28 UNTIL UNTIL end the BEGIN ... UNTIL loop, exit if flag true.
\ 29 EMIT  (07 for Bell ) to get audible output from the PC - ( : ) BELL 07 BEL ( ; )sends the bell signal to the speaker
\ 30 .S    .S is a non-destructive display of the stack, all of the items there
\ 31 .     . is similar to .S, prints out the top item only and as well consumes it; an easy way to get rid of the top stack item
\ 32 >R    >R takes an item from DSTACK and moves to the RSTACK;
\ 33 R>    R> does the opposite: take item from RSTACK move it to the top of DSTACK
\ 34 ( and ) These 2 brackets are used to put explanations within the code, and this is ignored by Forth, similar to \
\ ----- it seems this covers all of the words needed for this little sandbox application
\
\ This application VFXTESTAPP has not been programmed optimally - but this was not the target - beginner's code for beginners
\ Links to further information: see WORD file and PDF version of this text
\
: Name ." Hello Forth World " ;
Page CR Name 1000 ms
sos \ send SOS and then start the counter
counter \ start COUNTER

```

```

\ The Forth Machine:  Data stream coming in on the left, THE STACK, Return Stack, Variables in memory, other memory, route to screen.
\                      Digital_In and Digital_Out are for later, the same applies to Memory
\ The Forth Machine looks very complicated - but people actually use a similar model every day at the desk, work coming in, execute, results out
\
\   DIGITAL_IN          TOP_OF_STACK  TOP_OF_RS          x7          DIGITAL_OUT
\   ( for later )      DS-1           RS-1              x6          ( for later )
\
\   DS-2              RS-2              x5
\   DS-3              RS-3              x4
\   DS-4              RS-4              PSWI         x3
\   DS-5              RS-5              OUTP         x2
\   DS-6              RS-6              IN            x1
\   Token8 T7 T6 T5 T4 T3 T2 T1 T0  FFFF (-7)  FFFF          ANI          x0          Hello Forth World
\
\   Terminal_Input_Buffer_Contents  DATA STACK  RETURN_STACK  VARIABLES  MEMORY  TO_SCREEN
\

```

ExMark October 2016

MPE_VFX_LITE – download, installation and test

The image shows a screenshot of the MPE website and a Windows file dialog box. The website is titled "Downloads" and features a green header with the text "MPE - More real, less time" and contact information. The navigation menu includes Home, Software, Hardware, Consultancy, Training, Books, and Downloads. The main content area is divided into sections for "Cortex-Mx Lite compiler" and "MSP430 Lite compiler", each with links for instructions, getting started, and windows installer. A sidebar on the left lists various sections like "Main page & What's new?", "News, Gossip and Rumour", and "MPE Software".

Overlaid on the website is a Windows file dialog box titled "Opening xMSP430Lite.exe". The dialog displays the file "xMSP430Lite.exe" as a "Binary File (16.6 MB)" downloaded from "http://soton.mpeforth.com". It asks "Would you like to save this file?" and provides "Save File" and "Cancel" buttons.

In the background, a Windows Explorer window is visible, showing a folder structure for "MPE VFX Forth ARM" and "MPE VFX Forth ARM Cortex Cross Comp". The folder contents include "MPE VFX Forth Evaluation", "MPE VFX Forth MSP430 Cross Compiler", "AIDE PDF Manual", "AIDE Shell", "ANS Forth Specification (HTML)", "MSP430 Compiler PDF manual", "MSP430 Lite Compiler", "MSP430 Lite Target PDF manual", and "XC7 Compiler PDF manual".

Go to www.mpeforth.com and have a look around. Have a look at the Press section and see what is new.

This website is in the state of transition, it has been updated recently.

Open the tab downloads <http://www.mpeforth.com/arena.htm>. The link we want is the MSP430 Lite compiler and the link Windows installer.

I suggest you create a new folder MPEFORTH first.

Download into this folder and execute xMSP430lite.exe.

From the files there there are look at: AIDE Manual and start AIDE Shell

In the Utilities window you start ForthEd2 and Forth Console.

Resort the windows and you will have the same display as further down:

The AIDE Window with the 4 other windows :

Power Term – not need for now, so switch it off in Utilities.

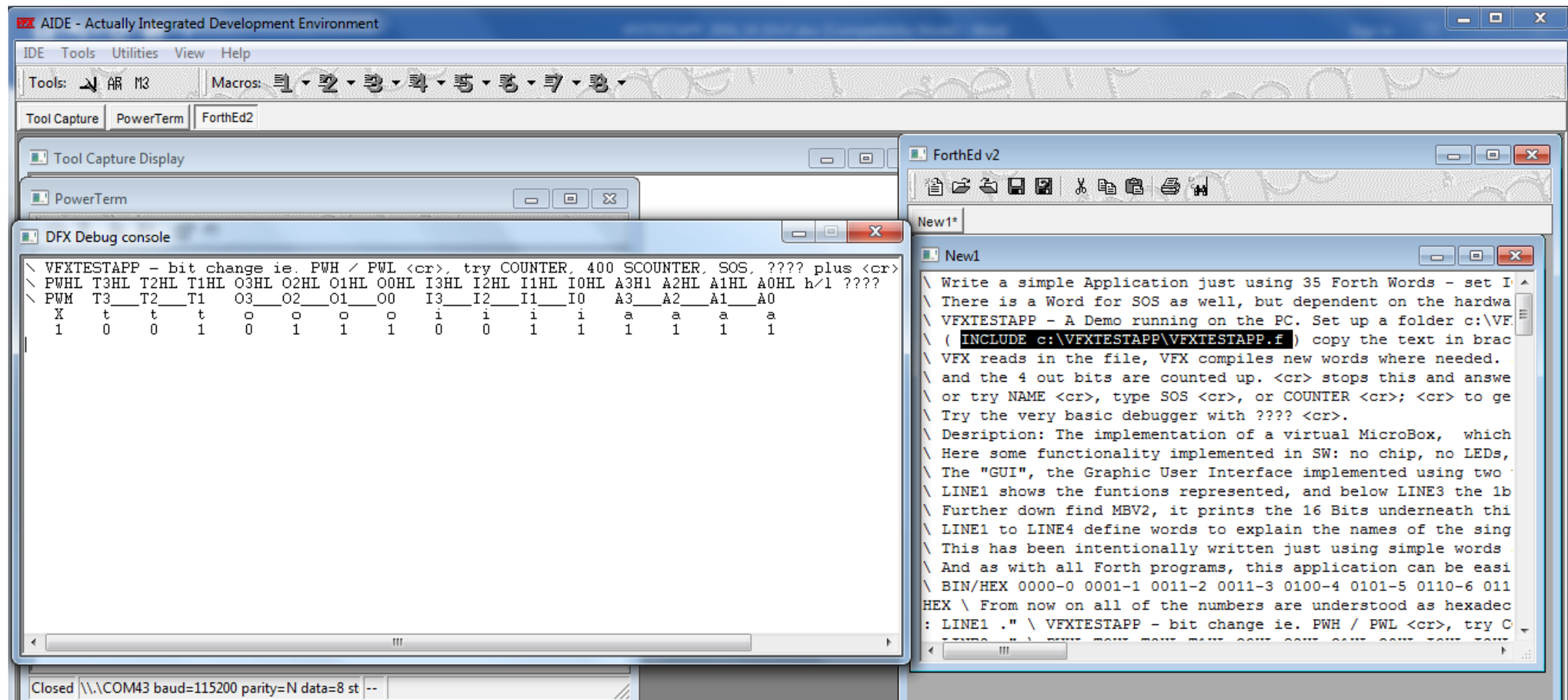
Tool Capture Display: this for later so can be disabled as well.

ForthEd2 our very basic ASCII editor.

And most importantly the **Forth Console**, this shows the VFX Forth software, here running VFXTESTAPP; and you can see part of the file in ForthEd2.

There will be a download location to be defined. For now it is <http://wiki.forth-ev.de/doku.php/en:projects:a-start-with-forth:start>

More explanations in the text of the application.



You can as well download a smaller size version of the AIDE Manual from the Forth-eV location if you wish - enjoy A Start in Forth