noForth

13nov2025

Using the library in noForth t

noForth t with USB has a large source code library in Flash ROM. The library consists of CHAPTERS containing program code. You can view the library with these two words:

```
CHAPTERS  \ List the names of all chapters.
LOOK ccc  \ Display the contents of the chapter named ccc.
```

Since we are not sure whether these two words are already in noForth, we execute the following code:

```
need CHAPTERS
need LOOK
```

`NEED ccc` \ Load chapter ccc from the library.
The special feature of NEED ccc is that chapter ccc is only loaded if ccc does not yet exist in forth.

Examples with CHAPTERS and LOOK

```
@)chapters
 COPYRIGHT  VERSION  UART  USB  TRY  ADD RESTORE-LIB  LOOK  DISPLAY
 VIEW  OPEN-LIB  WIPE-LIB  CLOSE-LIB  CHAPTER  .STACK  ALPHA -CHAPTERS
 NEED(  ()PIN  LOCK-PIN  48MHZ  125MHZ  132MHZ  250MHZ  38K4 115K2
 460K8  921K6  ()BAUD  [DATA  -ROT  ROLL  2TUCK  2ROT  -2ROT  ON
     ... etc.
```

```
@)look .s
\ .S
v: forth definitions
: .S    ( -- )  \ Non-destructive display of data stack
    ?stack (.) space
    depth false
    ?do  depth i - 1- pick
        base @ hx 0A = if . else u. then
    loop ;

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

Scrolling is done with the space bar; any other key stops the display.

What does NEED ccc actually do?

```
1) NEED searches for ccc in Forth.
2) If ccc is found, nothing else happens.
3) If ccc is not found in Forth, RUN ccc is executed.
   RUN ccc always loads chapter ccc, even if ccc already exists.
```

`s" ccc" NEEDED` is identical to NEED ccc. It is useful if you want to load something from the library from a forth definition.

There are chapters that do not compile code but only execute code, such as scripts, for example. In such cases, it does not matter whether you use NEED ccc or RUN ccc. An example of such a chapter is:

```
@)look 250mhz
\ 250MHZ
dm 250      0 cfg 2 +  h!   \ Set frequency in MHz
4 cfg @ abs 4 cfg !         \ Make sure to (re)start the second image if
config                      \ Test new configuration

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

`PIN` or `()PIN` ?
Some chapters need one or more arguments on the stack when they are loaded.
`CHAPTERS` displays those names with `()` in front of them. That is only a warning. `()` is NOT part of the real name.

```
@)3 run pin
Test S? 8  OK.0
```

```
@)need look  OK.0
@)look pin
\ PIN
( GPIO -- ) \ Change GPIO pin for S?
need [if]
depth 0= [if]  abort  [then]
dup dm 30 2 within [if]  drop  dm 24  [then] \ Invalid switch pin?
0 cfg c!            \ GPIO-xx for S?
4 cfg @ abs 4 cfg ! \ Make sure to (re)start the second image
config
cr .( Test S? ) s? .

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```