Library overview

Note that the stack comments added is the stack while loading the mentioned source code chapter. Library stack comments, some library chapters need data in the stack to configure the chapter. Below is an overview of the used stack comments:

name = String to be parsed from the input stream
ccc<ch> = String to be parsed delimited by the character 'ch'
a = Address
a b c etc. = Anonymus stack items
ch = ASCII character number
d = Double number
+n = Positive number
n = Signed number
u = Unsigned number
f = Flag (0 or -1)
pin = GPIO pin number
i*x j*x = Undefined stack contents

The basic word for using the library are:

CHAPTERS = Show all available code chapters
RUN (ccc --) = Interpret the chapter "ccc" using noForth
NEED (ccc --) = Check if code chapter "ccc" was already loaded, when not add that chapter to noForth

Tools

Tools function	**Description**
COPYRIGHT	Copyright message
VERSION	Library version info
UART	Switch back to UART interface
USB	Switch to USB CDC-interface (if any)
TRY ADD	Two pseudonyms for NEED & RUN

Tools function	Description
RESTORE-LIB	Restore library pointers
LOOK	Show source code belonging to “name”
DISPLAY	Pseudonym for LOOK
VIEW	Pseudonym for LOOK
OPEN-LIB	Open this library
WIPE-LIB	Erase this library
CLOSE-LIB	Close this library
CHAPTER	Add new source chapter to this library
.STACK	Show stack comment for CHAPTER if any
ALPHA	Show library in alphabatic order
-CHAPTERS	Show library in reverse order
NEED(	Multi name version of NEED
PIN (pin --)	Change ‘pin’ number for S?
LOCK-PIN	Disable S? (leave always true)
48MHZ	Change clock to 48 MHz
125MHZ	Change clock to 125 MHz
132MHZ	Change clock to 132 MHz
250MHZ	Change clock to 250 MHz
38K4	Change baudrate to 38400 baud
115K2	Change baudrate to 115200 baud
460K8	Change baudrate to 460800 baud
921K6	Change baudrate to 921600 baud
BAUD (u --)	Change baudrate to baudrate u from stack
[DATA	Inline data structure for colon definitions
-ROT	Rotate top item to third on the stack
ROLL	Move x-th item on the stack to the top
2TUCK	Copy d2 below d1
2ROT	Rotate third item to top of the stack
-2ROT	Rotate top item to third on the stack
ON OFF	Set or clear the cell at ‘addr’
2!	Store the double dx at ‘addr’
2@	Read the double dx from ‘addr’

Tools function	Description
0>	‘f’ is true when n is signed greater then zero
ARSHIFT	Arithmetic right shift by +n positions
D-	Subtract d2 from d1 leaving d3
M+	Add n to d1 leaving d2
DLSHIFT DRSHIFT	Double logical left & right shift
SM/REM	Symetric division with remainder
2LOG	Calculate binary logarithm y of u
CHARS	Add character address calculation
ERASE	Erase +n bytes from addr
J	Nested loop indexes J and K
PAD	Scratch pad area of 32 bytes
SOURCE	Input source manipulation words
CASE	The CASE statement word set
UNUSED	Leave free dictionary space
WORD	Parse the string delimited by ch leave it at a
[COMPILE]	Compile the word behind it always
ABORT"	Error message with inline string
[IF]	Interactive control structure word set
FORTH-WORDLIST	Compilation wordlist manipulation
GET-ORDER	Manipulate search order word set
SEARCH-WORDLIST	Search for wordname in given wordlist id
-TRAILING	Cut trailing spaces from a string
.S	Non-destructive display of data stack
[DEFINED]	Check if a word exist
[UNDEFINED]	Check if a word not exist
CRC	RP2040 style CRC generator
RANDOM	32-bit pseudo random Marsaglia generator, including CHOOSE (u1 – u2)
THENS	Close all open IFs
.BYTE	Print byte number in hexadecimal
.HEX	Print number unsigned in hexadecimal
PCHAR	Convert all data to printable characters

Tools function	Description
MANY	Redo current input line until a key was hit
STOP?	Leave flag when a key was pressed, hold on a space, abort on Esc.
RECUR	For use with for-next, it manipulates the index using keys from the keyboard
DMP	Simple dump routine, needs address only
DUMP	Classic Forth dump tool
WORDS	Show words in top vocabulary
@NAME	Read counted string from a header
>NFA	Convert an address to name field address
?TEXT	Search for inline text string
?HEAD	Check for a valid header at given address
.DATA	Print data word as chars and hexadecimal
SEE	noForth decompiler
.VOCS	Show all present vocabularies
.SHIELDS	Show all present shields
TOOLS\	Add basic noForth tool set
LARGE-TOOLS\	Add extended noForth tool set
ALLWORDS	16 times WORDS in noForth using kangaroo method
MEMMAP	Show noForth memory map
.CFG	Show noForth configuration
LAST\	Remove code behind the last present shield
-FLY	Make all control structures interactive
COMMACODE CM	Build assembler less code definitions
DAS\	Complete RP2040 disassembler
INSPECT\	Universal decompiler/disassembler
ASM\	noForth RP2040 T(humb) assembler
+ASM\	RP2040 assembler extension
PIO\	RP2040 PIO (dis)assembler v2
PIOBASE\	Minimal PIO control wordset v2
-LITERAL	Literal compiler, base addr. and offset

Tools function	Description
$VARIABLE	String manipulation word set
-TAIL -HEAD	Cut characters from a string
BITARRAY	Bit array word set, for compact on/off noting and operators to handle this array
*COPY	Copy any bit array to another, etc.
COUNT*	Count the bits set in given bit array
*UP?	Get the bit number of the lowest bit set & true and clear that bit, otherwise leave false
(?	Debugging through run-time stack check
?TASK	Check if a task is valid
TASK	Add, start & change background tasks
TASKS	Multitasker tools wordset
STK	Address stack cells of background tasks
LOCK	Semaphores, tools for sharing devices
SPINLOCK	Add spinlock words MINE? and YOURS
MINE?	Add spinlock words MINE? and YOURS
TASKER\	Add complete multitasker wordset
SET-FREQ	Set CPU frequency from 12MHz to 400MHz Note: a clock frequency above 300 MHz does not work on all RP2040 chips.
TESTER\	Add adapted Hayes test suite
TRACER\	Very basic number tracer with independent number conversion (.B .DUMP etc.)
IMAGE	Generate Intel-Hex from current boot image It can be converted to a UF2 with a small Win32Forth program
UNIT	Add Ullrich Hoffman’s UNIT structure
MEASURE\	Measure timing of code parts
CDC\	Single USB CDC driver runs in two tasks of the multitasker
CDC0\	Dual USB CDC driver that runs in three tasks of the multitasker. This part runs on core-0.
CDC1\	Second part of dual CDC driver, that runs on core-1
HARDFAULT	A hardfault handler that gives more details

Hardware

Hardware function	Description
CORE\	Add inter core communication
CORE1\	A blinker running on core-1
CORE0	A counter running on core-0
CORE1	A counter display progran running on core-1
BUTTON	A switch demo running on one core
RESPONSE	Inter core demo with the button program
BLINK	Simple GPIO25 flashing LED demo
ADC	Using the onboard ADC
BOOTKEY?	Reuse BOOTSEL switch for Forth input
TEMPERATURE	Use the built-in temperature sensor demo
PWM-ON (pin --)	Onboard dual PWM usage example on GPIO & GPIO + 1
ALARM	Using the alarm registers as interval timer
KHZ>	Convert wanted spi-clock to divider values
LOOPBACK	Loopback mode for spi-0 on/off
SPI\ (rx-pin 0|1 --)	spi-0 or spi-1 driver
I2C\ (clock sda-pin --)	Built-in I2C on all clock speeds
I2C-SLAVE\ (sda-pin --)	I2C slave implementation, needs wanted SDA pin number on the stack
IO-SLAVE	I2C PCF8574 style slave driver
COUNTER	I2C slave demo (master code part) set clock frequency & SDA pin number on the stack
MDMP	I2C slave demo (master code part) set clock frequency & SDA pin number on the stack
MEM-SLAVE	Implements an I2C memory slave, shares a memory part over the I2C-bus
>PCF8574	PCF8574 driver primitives
RUNNER	A few PCF8574 demo drivers
24C02\	Small I2C-eeprom code example
EEPROM	An eeprom code example
DEV?	Check if a device is present on the I2C-bus
SCAN-I2C	An i2c bus scanner
PCF8591\	I2C analog input & output using a PCF8591

Hardware function	Description
LM75\	I2C temperature measuring with LM75
WS2812	Control two WS2812 LEDs separately
WS2812START\	Self starting dual WS2812 PIO driver
WS2812MULTI\	Self starting dual WS2812 PIO driver using the multitasker
OLED character sets	
||	Read & compile 16-bit character row
|	Read & compile 8-bit character row
'GRAPH	Graphic characters of 8x4 pixels
'SMALL	Small characters set of 8x5 bits
'THIN	A 16x6 thin character set
`BOLD	A 16x8 bold character set
OLED SPI & I2C driver parts	
{OLED	OLED driver for GPIO16 and SPI-0
OLED-SPI\	Primitive SPI OLED text driver
I2C-OLED\	Primitive I2C OLED text driver
OLED screen control	
&CR	Add a scrolling display redefining the words &CR & &PAGE
THIN	Output 16x6 thin characters
BOLD	Output 16x8 bold characters
SMALL	Output 8x5 small characters
GRAPHIC	Output 8x4 graphic characters
OLED demo's need I2C or SPI driver loaded in front of it	
SMALL-DEMO	Small character scrolling demo
SCROLL-DEMO	Thin character scrolling demo
MIXED-DEMO	Mixed character size scrolling demo
THIN-DEMO	Large thin character size scrolling demo
BOLD-DEMO	Large bold character size scrolling demo
GRAPHIC-DEMO	Display graphic character set
EGEL-DEMO	Moving hedgehog demo
Completely functional SPI and I2C examples for Pico-kit	
SPI-OLED-DEMO\	Add all demos for an SPI OLED screen
I2C-OLED-DEMO\	Add all demos for an I2C OLED screen

Hardware function	**Description**
OLED-APP\ (name --)	Self-starting OLED app, add it to automate previous demo examples
ST7789\ (pin --)	240 x 280 TFT driver using SPI from pin onwards (needs 5 pins)
ST7789DEMO\	Add two basic color demo’s
MOVIE\	Movie player for ST7789
WRITEMOV\	Write movie’s to Flash memory
BAMBOE\ (pin #bamboe --)	Parallel output driver with serial input

PIO

PIO function	Description
BIT-TOGGLE1 (sm pio pin --)	Simple but flexible bit toggle
BIT-TOGGLE2 (pin --)	More advanced bit toggle on SM0 & PIO0
BIT-TOGGLE3 (pin --)	Idem 2 but with use of side-set & set
IN&OUT1 (pin --)	Use GPIO to activate GPIO+1
IN&OUT2 (pin --)	Toggle GPIO+1 using MOV, only
IN&OUT3 (pin --)	Toggle GPIO+1 using WAIT, & MOV,
IRQ-1 (pin --)	Two communicating programs using IRQ
IRQ-2 (pin --)	Idem 1, but using wrap function
MUSIC-0 (pin --)	Frequency generation using 4 GPIO's on 4 SM's uses clone
ON&OFF-1 (pin --)	Example of PIN? and MOV, toggles GPIO+1
ON&OFF-2 (pin --)	Example of PIN? and MOV, GPIO+1 on/off
ON&OFF-3 (pin --)	Idem 2, example of PIN? WAIT, and MOV,
PWM-1 (sm pio pin --)	1000 Hz PWM, range 0 to 100
PWM-2 (sm pio pin --)	1000 Hz PWM, range 0 to 200
PWM-3 (sm pio pin --)	1000 Hz PWM, range 0 to 400
PWM-4 (sm pio pin --)	10000 Hz PWM, range 0 to 400
ROTARY-0 (pin --)	Rotary encoder on GPIO to GPIO+3
ENCODER-DEMO	Encoder demo on GPIO26 to GPIO29
SPI-0 (sm pio clock pin --)	SPI I/O using three GPIO pins, clock in Khz 8 data bits and autopull & autopush
SPI-1 (sm pio clock pin --)	SPI I/O using four GPIO pins, clock in Khz 8 data bits, PULL, and autopush
SPI-2 (sm pio clock pin --)	SPI I/O using four GPIO pins, idem 2 but slightly different Forth usage
SPI-3 (sm pio clock pin --)	SPI I/O using four GPIO pins, the chip select is done by the Forth program
SPI-4 (sm pio clock pin --)	SPI out using four GPIO pins
SPI-5	SPI I/O using GPIO26 to GPIO29
UART-0	115K2 UART output on GPIO26
UART-1	Dual UART output on GPIO26 = 115K2 & GPIO27 = 38K4 using clone
UART-2	115K2 UART output on SM0 and GPIO26 & input on on SM1 and GPIO27

PIO function	Description
UART-3	460K8 UART TX and RX on GPIO26 & GPIO27 adding a simple chat function
UART-4	460K8 alternative UART for noForth
UART-5	9600 baud alternative UART for noForth
WS2812-0 (pin --)	Single WS2812 driver at GPIO from stack
WS2812-1 (sm pio pin --)	Single WS2812 driver demo & GPIO25 led flasher on two state machines
WS2812-2	Multiple WS2812 driver demo on GPIO23 & GPIO28, on two state machines using clone
WS2812-3	Multiple WS2812 LED on GPIO28 alternative driver demo, with Forth demo program & special functions
WS2812-4	Multiple WS2812 LED on GPIO28 precompiled PIO-program, with Forth demo program using tasker & special functions
CARS\	Highway simulation on a string of WS2812 leds (runs in the multitasker!)
DHT22\	DHT22 moisture and temperature sensor