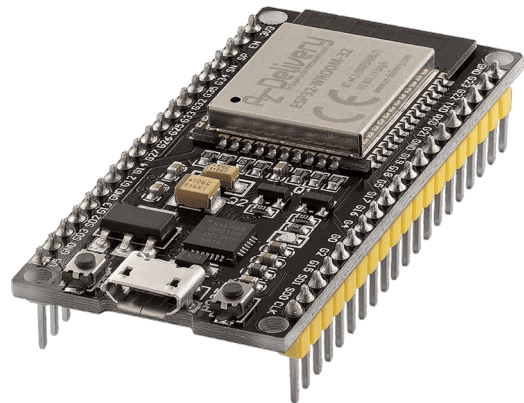# String Library for ESP32forth

**ESP32forth** is a powerful forth tool set created by **Bradley D Nelson** and the late **Dr. Hanson Ting** for the low cost ESP32 module. This document is a word glossary for a Portable String Library created by Mark Wills in 2014 and now adapted for ESP32forth by Bob Edwards in 2022.

The headings and words in the following table are in alphabetic order to speed up searches. Use ctrl F to search for a specific word. Immediate words (that run at compile time) are shown in red. The right-hand column indicates the ESP32forth vocabulary in which the word is located.

Any contributions welcome via the [Forth2020 forum](#) on Facebook

*ESP32 is a series of low-cost, low-power system on a chip microcontrollers with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 series employs either a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations, Xtensa LX7 dual-core microprocessor or a single-core RISC-V microprocessor and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules.*

## Modification to the ESP32forth Source Code

The U< word is needed to support this library. Add the following line to the ESP32forth source code underneath the definition for 0<> :-

X("U<", ULESS, tos = (ucell_t) *sp < (ucell_t) tos ? -1 : 0; --sp) \

## Strings

| -rot$ | ( -- ) ( ss: s3 s2 s1 -- s1 s3 s2) | Rotates the top three string to the right | forth |
|---|---|---|---|
| :=" | ( $Caddr "string" -- ) | Assigns the string "string" to the string constant. e.g. msg :=" hello mother!" N.B. Don't use in a definition only interpreted | forth |
| .$ | ( -- ) ( ss: str -- ) | Pop and display the topmost string from string stack | forth |
| .$const | ( $Caddr -- ) | Displays the string constant. e.g. fred .$const | forth |
| +$ | ( -- ) ( ss: s1 s2 -- s2+s1) | Replaces the top most two strings on the string stack with their concatenated equivalent | forth |
| ==$? | ( -- flag ) ( ss: -- ) | Performs a case-sensitive comparison of the | forth |

| | | topmost two strings on the string stack, returning true if their length and contents are identical, otherwise returning false | |
|---|---|---|---|
| **>$** | ( $Caddr -- ) | Moves a string constant to the string stack | forth |
| **>$const** | ( $Caddr -- ) ( ss: str -- ) | Move top of string stack to the string constant | forth |
| **$.s** | ( -- ) ( ss: -- ) | Non-destructively displays the string stack | forth |
| **$"** | ( "string" -- ) | Pushes a string directly to the string stack.<br> e.g. $" hello world" .$ | forth |
| **$>n** | ( -- n ) ( ss: str -- ) | Interprets the topmost string as a number, returning its value on the data stack as a signed integer | forth |
| **$const** | ( max_len "name" -- ) | Creates a string constant. When "name" is referenced the address of the max_len field is pushed to the stack.<br>  e.g. 100 string msg<br>A $const consists of a 'maxlength' long, 'currentlength' long followed by the actual characters. No termination character. So the address of the first char is msg 2 cells + | 'forth |
| **$trim** | ( -- ) ( ss: s1 -- s2 ) | Remove both leading and trailing spaces from s1, resulting in s2 | forth |
| **clen$** | ( $Caddr -- len ) | Given the address of a string constant, returns its length | forth |
| **depth$** | ( -- $sDepth) | Returns the depth of the string stack | forth |
| **drop$** | ( -- ) ( ss: str -- ) | Drops the top string from the string stack | forth |
| **dup$** | ( -- ) ( ss: s1 -- s1 s1) | Duplicates a string on the string stack | forth |
| **find$** | ( offset -- pos|-1 ) ( ss: s1 s2 -- s1) | Searches string s1, beginning at offset, for the substring s2. If the string is found, returns the position of the string relative<br> \ to the offset, otherwise returns -1 | forth |
| **findc$** | ( char -- pos|-1 ) ( ss: -- ) | Returns the first occurance of the character char in the top string. The string is retained. Returns -1 if the char is not found | forth |
| **lcase$** | ( -- ) ( ss: STR -- str) | On the topmost string, converts all upper case characters to lower case | forth |
| **left$** | ( len -- ) ( ss: str1 -- str1 str2) | The leftmost len characters are pushed to the string stack as a new string. The original string is retained | forth |
| **len$** | ( -- len ) ( ss: -- ) | Returns the length of the topmost string | forth |

| | | | |
|---|---|---|---|
| **ltrim$** | ( -- ) ( ss: s1 -- s2 ) | Removes leading spaces from s1, resulting in s2 | forth |
| **maxLen$** | ( $Caddr -- max_len ) | Given the address of a string constant, returns its maximum length | forth |
| **mid$** | ( start len -- ) ( ss: str1 -- str1 str2) | The word mid$ produces a sub-string on the string stack, consisting of the characters from the topmost string starting at character start and ending at character end | forth |
| **n>$** | ( n -- ) ( ss: -- str ) | Pushes the signed number on the data stack to the string stack | |
| **nip$** | ( -- ) ( ss: s1 s2 -- s2) | Remove the string under the top string | forth |
| **over$** | ( -- ) ( ss: s1 s2 -- s1 s2 s1) | Move a copy of s1 to top of string stack | forth |
| **pick$** | ( n -- ) ( ss: -- strN) | Given an index into the string stack, copy the indexed string to the top of the string stack.<br>  0 $pick is equivalent to $DUP<br>  1 $pick is equivalent to $OVER etc. | forth |
| **replace$** | ( -- pos ) ( found: ss: s1 s2 s3 -- s4 not found: s1 s2 -- s1 s2) | In string s2 find s3 and replace with s1, resulting in s4. If a replacement is made, the starting position of the replacement is returned, otherwise -1 is returned | forth |
| **rev$** | ( -- ) ( ss: s1 -- s2 ) | Reverse topmost string on string stack | forth |
| **right$** | ( len -- ) ( ss: str1 -- str1 str2) | The rightmost len characters, pushed to the string stack as a new string. The original string is retained | forth |
| **rot$** | ( -- ) ( ss: s3 s2 s1 -- s2 s1 s3) | Rotates the top three string to the left | forth |
| **rtrim$** | ( -- ) ( ss: s1 -- s2 ) | Removes trailing spaces from s1, resulting in s2 | forth |
| **swap$** | ( -- ) ( ss: s1 s2 -- s2 s1) | Swaps the top two string items on the string stack | forth |
| **ucase$** | ( -- ) ( ss: str -- STR) | On the topmost string, converts all lower case characters to upper case | forth |

# String Stack Internals

With two strings on the stack, the contents is:-

| |
|---|
| Length of string1 |
| s<br>t<br>r<br>I<br>n<br>g<br>1<br>padding if reqd |
| Length of string2 |
| S<br>t<br>r<br>I<br>n<br>g<br>2<br>padding if reqd |

← ($sp@) points here   ← Lowest address is top of stack

← Highest address

N.B. the length cells contain ( No. of chrs + length cell + any padding) in bytes