AP FORTH

AP Forth is a minimal, string-threaded Forth interpreter designed for the PIC16F18026 microcontroller (AP118), compiled with Microchip's XC8 toolchain under MPLAB X. It operates within tight constraints — roughly 5 KB of program ROM and 560 bytes of RAM — yet provides a surprisingly complete interactive programming environment over a serial terminal.

The interpreter uses 16-bit signed cells (matching XC8's native `int` on PIC16) and supports around 60 built-in primitives covering stack manipulation, integer arithmetic, bitwise logic, comparisons, serial I/O, direct SFR and RAM access via `C@` and `C!`, and three styles of flow control: `IF…ELSE…THEN` conditionals, `BEGIN…UNTIL/AGAIN` indefinite loops, and `DO…LOOP/+LOOP` counted loops with `I`, `J`, and `LEAVE`.

User-defined words are created with the standard `: name body ;` syntax. Definitions are stored as raw null-terminated text in a 384-byte RAM dictionary and re-interpreted on each call, with up to five levels of nesting supported. A non-recursive interpreter core avoids the compiled-stack aliasing pitfalls inherent to XC8's PIC16 code generation. Number literals can be entered in decimal, hexadecimal (`$` prefix), binary (`%` prefix), or as character constants (`'A'`), and the base can be switched at any time with `HEX` and `DECIMAL`.

Because `C@` and `C!` provide direct byte-level access to every SFR address, the interpreter doubles as a live hardware debugging and prototyping tool — you can toggle GPIO pins, configure peripherals, and probe registers without recompiling. Combined with `MS` for millisecond delays and `DO…LOOP` for iteration, it's straightforward to write interactive hardware test routines entirely from the serial prompt.

Serial is on RA4 (TX) and RA5 (RX). 9600bps N81.

Future enhancements include moving user words into flash memory, autostart and headless code compilation to improve code space usage.

Word Reference — PIC16F18026

String-threaded interpreter • 16-bit signed cells • XC8 / MPLAB X

Data stack: 16 deep	Return stack: 12 deep	Input buffer: 80 chars	Dictionary: 384 bytes

Stack Manipulation

Word	Stack Effect	Description
DUP	(n -- n n)	Duplicate top of stack.
DROP	(n --)	Remove top of stack.
SWAP	(a b -- b a)	Swap top two items.
OVER	(a b -- a b a)	Copy second item to top.
ROT	(a b c -- b c a)	Rotate third item to top.
NIP	(a b -- b)	Remove second item.
TUCK	(a b -- b a b)	Copy top item below second.
2DUP	(a b -- a b a b)	Duplicate top pair.
2DROP	(a b --)	Remove top pair.
DEPTH	(-- n)	Push current stack depth.

Arithmetic

Word	Stack Effect	Description
+	(a b -- a+b)	Add.
-	(a b -- a-b)	Subtract (a minus b).
*****	(a b -- a*b)	Multiply.
/	(a b -- a/b)	Signed integer divide.
MOD	(a b -- a%b)	Signed integer remainder.
1+	(n -- n+1)	Increment by one.
1-	(n -- n-1)	Decrement by one.
2*	(n -- n*2)	Multiply by two.
2/	(n -- n/2)	Divide by two (signed).
NEGATE	(n -- -n)	Two’s complement negate.
ABS	(n -- |n|)	Absolute value.
MAX	(a b -- max)	Larger of two values.
MIN	(a b -- min)	Smaller of two values.

Bitwise Logic

Word	Stack Effect	Description
AND	(a b -- a&b)	Bitwise AND.
OR	(a b -- a|b)	Bitwise OR.

Word	Stack Effect	Description
XOR	(a b -- a^b)	Bitwise exclusive OR.
INVERT	(n -- ~n)	Bitwise complement (all bits flipped).
LSHIFT	(n u -- n<<u)	Logical left shift by u bits.
RSHIFT	(n u -- n>>u)	Logical right shift by u bits.

Comparison

Word	Stack Effect	Description
=	(a b -- flag)	True (–1) if a equals b, else false (0).
<>	(a b -- flag)	True if a does not equal b.
<	(a b -- flag)	True if a is less than b (signed).
>	(a b -- flag)	True if a is greater than b (signed).
0=	(n -- flag)	True if n is zero.
0<	(n -- flag)	True if n is negative.
0>	(n -- flag)	True if n is positive.

Input / Output

Word	Stack Effect	Description
EMIT	(c --)	Send character c to the serial terminal.
KEY	(-- c)	Wait for and push one character from terminal.
KEY?	(-- flag)	True if a key is available (non-blocking).
.	(n --)	Print n in the current base, followed by a space.
.HEX	(n --)	Print n in hexadecimal, followed by a space.
."	(--)	Print the string that follows up to the closing ". Example: ." Hello"
CR	(--)	Print a carriage-return / line-feed.
SPACE	(--)	Print one space character.
SPACES	(n --)	Print n space characters.
BL	(-- 32)	Push ASCII code for space (32).
.S	(--)	Non-destructively display the entire data stack.

Memory Access

Word	Stack Effect	Description
@	(addr -- n)	Fetch 16-bit cell from addr.
!	(n addr --)	Store 16-bit cell n at addr.
C@	(addr -- byte)	Fetch single byte from addr.
C!	(byte addr --)	Store single byte at addr.

Word	Stack Effect	Description
+!	(n addr --)	Add n to the 16-bit cell at addr.

Return Stack

Word	Stack Effect	Description
>R	(n --) (R: -- n)	Move top of data stack to return stack.
R>	(-- n) (R: n --)	Move top of return stack to data stack.
R@	(-- n) (R: n -- n)	Copy top of return stack to data stack.

Conditionals

Word	Stack Effect	Description
IF	(flag --)	If flag is true (non-zero), execute the following words. If false, skip to ELSE or THEN.
ELSE	(--)	Marks the start of the false branch. Reached only when IF's flag was true (skips to THEN).
THEN	(--)	Terminates an IF or IF…ELSE structure.

Indefinite Loops

Word	Stack Effect	Description
BEGIN	(--)	Mark the start of a loop.
AGAIN	(--)	Unconditionally jump back to BEGIN (infinite loop).
UNTIL	(flag --)	If flag is false (0), loop back to BEGIN. If true, exit loop.

Counted Loops (DO…LOOP)

Word	Stack Effect	Description
DO	(limit start --)	Begin a counted loop from start to limit–1. If limit equals start the body is skipped entirely (zero-trip).
LOOP	(--)	Increment the loop index by 1. If the index reaches the limit, exit; otherwise loop back to just after DO.
+LOOP	(n --)	Add n to the loop index. Exits when the index crosses the limit boundary (works for both positive and negative steps).
I	(-- index)	Push the current (innermost) loop index.
J	(-- index)	Push the outer loop index (for nested DO loops).
LEAVE	(--)	Exit the innermost DO…LOOP immediately.

System / State

Word	Stack Effect	Description
HEX	(--)	Set number base to 16.
DECIMAL	(--)	Set number base to 10.
BASE	(-- n)	Push the current number base.
TRUE	(-- -1)	Push true flag (–1).
FALSE	(-- 0)	Push false flag (0).

Word	Stack Effect	Description
MS	(n --)	Delay for n milliseconds.

Dictionary / Compiler

Word	Stack Effect	Description
:	(--)	Begin a new word definition. Syntax: : name body words ;
;	(--)	End a word definition (used inside : only).
WORDS	(--)	List all built-in and user-defined words.
FORGET	(--)	Remove the named word (and everything defined after it) from the dictionary. Syntax: FORGET name
BYE	(--)	Reset the microcontroller.

Number Literals

Word	Stack Effect	Description
123	(-- 123)	Decimal literal (default base 10).
$1A	(-- 26)	Hex literal ($ prefix forces base 16).
#99	(-- 99)	Decimal literal (# prefix forces base 10).
%1010	(-- 10)	Binary literal (% prefix forces base 2).
'A'	(-- 65)	Character literal (ASCII value of A).

Quick Examples

```
: blink  5 0 do led-on 500 ms led-off 500 ms loop ;
```
Define a word that blinks an LED 5 times (500 ms on, 500 ms off).

```
10 0 do i . loop
```
Print numbers 0 through 9.

```
: factorial  1 swap 1+ 1 do i * loop ;
```
Compute factorial: 5 factorial . prints 120.

```
$14 c@ $DF and $14 c!
```
Clear bit 5 of TRISC (set RC5 as output).

```
$1A c@ $20 or $1A c!
```
Set bit 5 of LATC (turn on LED on RC5).

```
: count  begin dup . 1- dup 0= until drop ;
```
Count down from n to 1 using BEGIN…UNTIL.

```
10 0 do i 5 = if leave then i . loop
```
Print 0 1 2 3 4, then exit early with LEAVE.

```
50 0 do i . 2 +loop
```
Print even numbers: 0 2 4 6 … 48.