noForth assembler notation

(january 2018)

noForth website

Registers

noForth	ti	
PC	R0	program counter
RP	R1	return stack pointer N.B.
SR	R2	status register
CG	R3	constant generator
SP	R4	data stack pointer N.B.
IP	R5	forth instruction pointer
W	R6	local noForth scratch register
TOS	R7	top of data stack
DAY	R8	local noForth scratch register
MOON	R9	local noForth scratch register
SUN	R10	local noForth scratch register
XX	R11	free
YY	R12	free
ZZ	R13	free
DOX	R14	noForth register for do-loops
NXT	R15	noForth register with 'next' address

Source addressing modes

noForth	ti
sp	R4
sp)	@R4
sp)+	@R4+
2 sp x)	2(R4)
1234 &	&1234
1234 #	#1234 (any number)
#-1 #0 #1 #2 #4 #8	special numbers

Destination addressing modes

noForth	ti
sp	R4
sp)	0(R4)
2 sp x)	2(R4)
1234 &	&1234
tos sp -) mov	is short for:
#2 sp sub tos sp) mov	SUB #2,R4 MOV TOS,0(R4)

Conditionals & Conditions

```
if,     ahead,  else,  then,
until,  begin,  again,
while,  repeat,
```

noForth		ti
=?	0=?	JNE/JNZ
<>?	0<>?	JEQ/JZ
<eq?		JL
>?		JGE
cs?	u<eq?	JNC
cc?	u>?	JC
pos?		JN

To be used before:

```
if, until, while,
```

Avoided name conflicts N.B.

noForth	ti
BIX (or XOR>)	XOR
BIA (or AND>)	AND

Examples

```
code ! ( x a -- )
sp )+ tos ) mov
sp )+ tos mov
next end-code

code DUP ( x -- x x )
tos sp -) mov
next end-code

code MIN ( x y -- z )
sp )+ w mov   tos w cmp
>? if,                  \ tos > w ? N.B.
       w tos mov
then,    next end-code

code LSHIFT ( x1 n -- x2 )

tos w mov
sp )+ tos mov
#0 w cmp
<>? if, begin,   tos tos add
                 #1 w sub
        =? until,
then,   next end-code
```

Cycles

adr	src	dest
r	0	1
))+ #	1	-
x)	2	4

Add 1 cycle when
PC is destination.