



[noForth website](http://noForth.org)

noForth assembler notation for MSP430

(october 2018)

27 instructions (16b, .B = 8b)

- ◆ 6 instructions with 1 operand
RRA RRC SWPB SXT PUSH CALL
- ◆ 12 instructions with 2 operands
MOV CMP ADD SUB ADDC SUBC DADD
BIT BIS BIC *BIA* (AND) *BIX* (XOR)
- ◆ 8 jump instructions with relative inline addresses
JNZ JZ JNC JC JN JGE JL JMP
- ◆ return from interrupt
RETI

Emulated instructions

```

BR      <addr> #   PC MOV
POP     RP )+   <dest> MOV
RET     RP )+   PC MOV
ADC     #0     <dest> ADDC
SBC     #0     <dest> SUBC
NOP     #0     CG MOV
TST     #0     <dest> CMP
DEC     #1     <dest> SUB
INC     #1     <dest> ADD
SETC    #1     SR BIS
CLRC    #1     SR BIC
SETZ    #2     SR BIS
CLRZ    #2     SR BIC
SETN    #4     SR BIS
CLRN    #4     SR BIC
EINT    #8     SR BIS
DINT    #8     SR BIC
INV     #-1    <dest> BIX
RLA     <dest> <dest> ADD
RLC     <dest> <dest> ADDC  etc.

```

16 Registers

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

Source addressing modes

<u>ti</u>	<u>noForth</u>
R4	sp
@R4	sp)
@R4+	sp)+
2(R4)	2 sp x)
&1234	1234 &
#1234 (any number)	1234 #
the six special numbers	#-1 #0 #1 #2 #4 #8

Destination addressing modes

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

Conditionals and Conditions

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

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

Use the question mark conditions before *if, until, while,*
ex.
0=? if, ... then,

Avoided name conflicts

<u>ti</u>	<i>noForth</i>
XOR	<i>BIX</i> <i>N.B.</i>
AND	<i>BIA</i> <i>N.B.</i>

Code examples

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

```
code c@ ( a -- ch )
  tos ) tos .b mov \ N.B.
  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

	src	dest
reg	0	1
))+ #	1	-
x)	2	4

Add 1 cycle when PC is destination.