MSP430 Lite Target Glossary – User WORDS of stand alone version - MCU is MSP430G2553

Microprocessor Engineering Limited

c Published by Microprocessor Engineering Copyright © 2003, 2005, 2007, 2009, 2013 Microprocessor Engineering Limited
MSP430 Lite Target v7.3
User manual Manual,
revision 7.3 22 June 2014
Software Software version 7.3

For technical support Please contact your supplier

For further information
MicroProcessor Engineering Limited
133 Hill Lane Southampton SO15 5AF UK
Tel +44 (0)23 8063 1441
Fax +44 (0)23 8033 9691
e-mail mpe@mpeforth.com
e-mail tech-support@mpeforth.com
web www.mpeforth.com

How to read the Glossary

Forth words are printed in upper case followed by a picture of the data stack, seperated by a backslash, the forth comment sign. This is done in the corosscompiler source to seperate the definition from a comment on it. Since the glossary is mostly generated automatic, this commenting stype is kept in this printing. There may be complex stack pictures. Only those forth words are given here, that are output by WORDS to Terminal.

4.2 Literal and flow of control

```
EXECUTE \ xt --
I \ --n
J \ --n
UNLOOP \ --
LEAVE \ --
```

4.3 Flash operations

```
FLERASE \ addr len --

C!F \ b addr --
!F \ w addr --
```

4.4 Digits and strings

```
DIGIT \ char base -- 0 | n true
/SRTING \ addr len n -- addr+n len-n
CMOVE \ source dest len --
CMOVE> \ source dest len --
FILL \ addr len char --
ERASE \ addr len --
S= \ addr1 addr2 count --flag
SKIP \ c-addr u char -- 'c-addr 'u
SCAN \ caddr u char -- caddr2 u2
COUNT \ addr --addr+1 len
(") \ --addr
UPC \ char -- charÕ
UPPER \ c-addr len --
PLACE \ c-addr1 u c-addr2 --
```

4.5 Arithmetic

4.5.1 Basics

```
1+ \ n -- n+1
2+ \ n -- n+2
1- \ n -- n-1
2- \ n -- n-2
2* \ n1 -- n2
U2/ \ n1 -- n2
2/ \ n1 -- n2
- \ n1 n2 -- n1-n2
+ \ n1 n2 -- n1+n2
NEGATE  \ n1 ---n1
ABS \ n1 --|n1|
DNEGATE \ d1 ---d1
DABS \ d1 --|d1|
D+ \ d1 d2 -- d3
D- \ d1 d2 -- d1-d2
S>D \ n -- d
D< \ d1 d2 -- t/f
D> \ d1 d2-- t/f
D0= \ d -- t/f
D= \ d1 d2 -- t/f
```

4.5.2 Multiplication

```
UM* \ u1 u2 -- ud
* \ n1 n2 -- n1*n2
M* \ n1 n2 -- d
```

4.5.3 Division

```
UM/MOD \ u32 u16 --urem uquot
SM/REM \ d n --rem quot
/MOD \ n1 n2 --rem quot
/ \ n1 n2 -- quot
MOD \ n1 n2 -- rem
MU/MOD \ ud1 u2 --u3 ud4
```

4.6 Logic

```
AND \ n1 n2 -- n3
OR \ n1 n2 -- n3
XOR \ n1 n2 -- n3
INVERT \ n1 --n2
```

4.7 Shifts

```
LSHIFT \ x count --xÕ
RSHIFT \ x count --xÕ
```

4.8 Return stack words

```
>R \ x -- ; R -- x
R@ \ --x ; R x -- x
R> \ --x ; R:x--
```

4.9 Comparisons

```
= \ n1 n2 -- flag
<> \ n1 n2 -- flag
0<> \ n -- flag
0= \ n -- flag
0< \ n -- flag
0> \ n -- flag
U< \ n1 n2 -- flag
U> \ n1 n2 -- flag

< \ n1 n2 -- t/f
> \ n1 n2 -- t/f
<= \ n1 n2 -- t/f
>= \ n1 n2 -- t/f

MIN \ n1 n2 -- min(n1,n2)
MAX \ n1 n2 -- max(n1,n2)
```

4.10 Stack primitives

```
OVER \ n1 n2 -- n1 n2 n1
2OVER \ n1 n2 n3 n4 -- n1 n2 n3 n4 n1 n2
DROP \ n1 --
2DROP \ n1 n2 --
SWAP \ n1 n2 – n2 n1
2SWAP \ n1 n2 n3 n4 -- n3 n4 n1 n2
DUP \ n1 -- n1 n1
2DUP \ n1 n2 -- n1 n2 n1 n2
?DUP \ n1 -- n1 [n1]
NIP \ n1 n2 -- n2
TUCK \ n1 n2 -- n2 n1 n2
PICK \ nn..n0 n -- nn..n0 nn
ROT \ n1 n2 n3 -- n2 n3 n1
-ROT \ n1 n2 n3 -- n3 n1 n2
C@ \ addr -- b
@ \ addr -- n
2@ \ addr -- d
C! \ b addr --
! \ n addr --
2! \ d addr --
+! \ n addr --
NOOP \ -- ; dummy
WITHIN \ n1|u1 n2|u2 n3|u3 -- flag
ON \ addr --
OFF \ addr --
BOUNDS \ addr len -- addr+len addr
NAME> \ nfa -- cfa
>NAME \ cfa -- nfa
SEARCH-WORDLIST \ c-addr u wid --0|xt 1|xt -1
```

4.11 Portability words

```
CELL \ -- 2
ALIGNED \ addr --addrÕ
>BODY \ xt -- pfa
COMPILE, \ addr --
```

4.12 Defining words

```
DOES> \ C: colon-sys1 -- colon-sys2 ;
Run: -- ; R nest-sys --
  \ C: "<spaces>name" -- colon-sys ;
Exec i*x --j*x ; R -- nest-sys
CONSTANT \ x "<spaces>name" -- ; Exec --x
EQU \ x "<spaces>name" -- ; Exec --x
VARIABLE \ "<spaces>name" -- ; Exec --a-addr
USER \ u "<spaces>name" -- ; Exec --addr
DEFER \ Comp "<spaces>name" -- ; Run: i*x --j*x
```

MSP430 Lite Target Glossary – User WORDS of stand alone version - MCU is MSP430G2553

```
' <action> IS <deferredword>
['] <action> IS <deferredword>
```

4.13 Multitasker hook

```
PAUSE \ --
```

4.14 Reboot

```
REBOOT \ --
```

5.1 User variables

```
SELF \ addr
S0 \ -- addr
R0 \ -- addr
'TIB \ -- addr
#TIB \ -- n
>IN \ -- n
OUT \ -- n
DPL \ -- addr
OPVEC \ -- addr
IPVEC \ -- addr
PAD \ -- addr
```

5.2 System data

5.2.1 Constants

```
BL \ -- char
```

5.2.2 System variables and data

```
FENCE \ -- addr
DP \ -- addr
RP \ -- addr
XDP \ --addr
DISK-ERROR \ -- addr
```

5.3 Vectored I/O handling

(see user manual)

5.3.3 Generic I/O words

```
KEY \ --char
KEY? \ --flag
EMIT \ --char
CR \ --
SPACE \ --
SPACES \ n --
```

5.4 Laying data in memory

```
HERE \ -- addr
ORG \ addr --
ALLOT \ n --
RHERE \ -- addr
RALLOT \ n --
ROM \ --
RAM \ --
ALIGNED \ addr --addr'
ALIGN \ --
, \ x --
C, \ char --
```

5.5 Dictionary management

```
FIND \ c-addr -- c-addr 0|xt 1|xt -1
.NAME \ nfa --
CREATE \ --
<BUILDS \ --
```

5.6 String compilation

5.7 ANS words CATCH and THROW

(see user manual)

5.7.3 User words

```
CATCH \ i*x xt -- j*x 0|i*x n
THROW \ k*x n -- k*x|i*x n
?THROW \ flag throw- -- ;
ABORT" \ Comp "ccc<quote>" -- ; Run: i*x x1 --| i*x ; R j*x --| j*x
```

5.8 Formatted and unformatted i/o

5.8.1 Setting number bases

```
HEX \ --
DECIMAL \ --
```

5.8.2 Numeric output

```
HOLD \ char --
# \ ud1 -- ud2
#S \ ud1 -- ud2
<# \ --
#> \ xd -- c-addr u
D.R \ dn --
D. \ d --
. \ n --
U. \ u --
.R \ n1 n2 --
```

5.8.3 Numeric input

```
+DIGIT \ d1 n --d2
>NUMBER \ ud1 c-addr1 u1 --ud2 c-addr2 u2
NUMBER? \ $addr -- n1|d2|0
```

5.9 String input and output

```
." \ "ccc<quote>" --
$. \ c-addr --
ACCEPT \ c-addr +n1 --+n2
TYPE \ c-addr len --
```

5.10 Source input control

```
QUERY \ --
```

5.11 Text scanning

```
PARSE \ char "ccc<char>" --c-addr u
WORD \ char "<chars>ccc<char>" --c-addr
```

5.12 Miscellaneous

```
WORDS \ --
MOVE \ addr1 addr2 u --
DEPTH \ ??? -- +n
TESTAPP \ -- ; enless loop, reset to leave loop.
```

5.13 Wordlist control

(see user manual)

5.14 Control structures

```
DO \ C: -- do-sys ; Run: n1|u1 n2|u2 -- ; R -- loop-sys
?DO \ C: -- do-sys ; Run: n1|u1 n2|u2 -- ; R -- | loop-sys
LOOP \ C: do-sys -- ; Run: -- ; R loop-sys1 -- | loop-sys2
+LOOP \ C: do-sys -- ; Run: n -- ; R loop-sys1 -- | loop-sys2
BEGIN \ C: -- dest ; Run: --
AGAIN \ C: dest -- ; Run: --
UNTIL \ C: dest – ; Run: x --
WHILE \ C: dest -- orig dest ; Run: x --
REPEAT \ C: orig dest -- ; Run: --
IF \ C: -- orig ; Run: x--
THEN \ C: orig -- ; Run: --
ELSE \ C: orig1 -- orig2 ; Run: --
RECURSE \ Comp --
```

5.15 Target interpreter and compiler

```
POSTPONE \ Comp "<spaces>name" --
S" \ Comp "ccc<quote>" -- ; Run: --c-addr u
C" \ Comp "ccc<quote>" -- ; Run: --c-addr
LITERAL \ Comp x -- ; Run: -- x
[CHAR] \ Comp "<spaces>name" -- ; Run: -- char
[ \ --
] \ --
IMMEDIATE \ --
' \ "<spaces>name" --xt
['] \ Comp "<spaces>name" -- ; Run: --xt
[COMPILE] \ "<spaces>name" --
( \ "ccc<paren>" --
\ \ "ccc<eol>" --
", \ "ccc<quote>" --
IS \ "<spaces>name" --
EXIT \ R nest-sys --
; \ C: colon-sys -- ; Run: -- ; R nest-sys --
INTERPRET \ --
EVALUATE \ i*x c-addr u -- j*x
QUIT \ -- ; R i*x --
```

5.16 Startup

5.16.1 The COLD sequence

```
COMMIT \ xt|0 --
EMPTY \ --
COLD \ --
```

6. Time Delays

```
TICKS \ -- n
LATER \ n --n'
TIMEDOUT? \ n -- flag
MS \ n --
```

7. Debug tools

```
DUMP \ addr len --
.S \ i*x -- i*x
```

8. Compile source from AIDE

```
END-LOAD \ --
INCLUDE \ "<filename>" --
```

11.2 UART

```
CONSOLE0 \ -- addr
CONSOLE \ -- addr
```

TYPE \ c-addr len --

12. Ticker using watchdog timer

<TICKS> \ -- addr ; variable
TICKS \ -- n
LEDACTIVE \ -- addr
START-CLOCK \ --
STOP-CLOCK \ --

13.1 Basic port usage

(see user manual)
GREEN-ON \ -- ; P1.6
GREEN-OFF \ -- ; P1.6
RED-ON \ -- ; P1.0
RED-OFF \ -- ; P1.0

13.2 Port counting using interrupts

13.3 Simple ADC driver

(see user manual)

5.17 Kernel error codes

-1 ABORT
-2 ABORT"
-4 Stack underflow
-13 Undefined word.
-14 Attempt to interpret a compile only definition.
-22 Control structure mismatch -unbalanced control structure.
-121 Attempt to remove with MARKER or FORGET below FENCE in protected dictionary.
-403 Attempt to compile an interpret only definition.
-501 Error if not LOADing block.

3. MSP430G2553 start up

3.1 Magic addresses

$10FF CALBC1_1MHz
$10FE CALDCO_1MHz
$10FD CALBC1_8MHz
$10FC CALDCO_8MHz
$10FB CALBC1_12MHz
$10FA CALDCO_12MHz
$10F9 CALBC1_16MHz
$10F8 CALDCO_16MHz
$10F7 08 Size (bytes) of value data
$10F6 01 Tag

Port 1 is used as follows

P1.0 Green LED, high=on
P1.1 UART Rx
P1.2 UART Tx
P1.3 Button Switch input
P1.4 --
P1.5 --
P1.6 Red LED, high=on
P1.7 --

3.2 Start of Forth

3.3 Default Interrupt vectors

3.4 Reset values for user and system variables

(see user manual)

4. MSP430 definitions

4.1 Register usage
IP R0/PC
RSP R1/SP
    R2/CG1/SR
    R3/CG2
PSP R4
TOS R5
UP R6
LP (locals) R7
scratch R8..R13
codegen R14 temp
codegen R15 SR

Memory Map

.. peripherals
$0200 - $0400 RAM
$1000 - $10FF INFO 4 segments of 64 bytes flash
 .. nc
$C000 - $CFFF application flash (4K)
$D000 - $FFFF liteLP2553sa code (12287 bytes)

RAM downwards

$0400   (R0 @)
... returnstack, &64 ($40) bytes down
$03BC   (S0 @)
... datastack, &64 ($40) bytes down
$032C   (PAD)
... scrach pad
$02E0
...   terminal input buffer, &64 ($40) bytes up
$02A0   ('TIB @)
...   unused RAM, &40 cells up, variables
$024E   (RHERE)
...   system user variables
$0200

INFO segments
$1000 info-D
$1040 info-C   APPSTART forth system information.
$1080 info-B
$10C0 info-A   Segment A contains calibration data.

(M.Kalus, 10/2014)

MSP430 Lite Target Glossary – User WORDS of stand alone version - MCU is MSP430G2553

Alphanumeric List

```
: \ C: "<spaces>name" -- colon-sys ; Exec i*x --j*x ; R -- nest-sys
- \ n1 n2 -- n1-n2
-ROT \ n1 n2 n3 -- n3 n1 n2
, \ x --
; \ C: colon-sys -- ; Run: -- ; R nest-sys --
! \ n addr --
!F \ w addr --
?DO \ C: -- do-sys ; Run: n1|u1 n2|u2 -- ; R -- | loop-sys
?DUP \ n1 -- n1 [n1]
?THROW \ flag throw- -- ;
. \ n --
." \ "ccc<quote>" --
.NAME \ nfa --
.R \ n1 n2 --
.S \ i*x -- i*x
' <action> IS <deferredword>
' \ "<spaces>name" --xt
'TIB \ -- addr ; user variable
", \ "ccc<quote>" --
( \ "ccc<paren>" --
(") \ --addr
[ \ --
['] <action> IS <deferredword>
['] \ Comp "<spaces>name" -- ; Run: -- xt
[CHAR] \ Comp "<spaces>name" -- ; Run: -- char
[COMPILE] \ "<spaces>name" --
] \ --
@ \ addr -- n
* \ n1 n2 -- n1*n2
/ \ n1 n2 -- quot
/MOD \ n1 n2 --rem quot
/SRTING \ addr len n -- addr+n len-n
\ \ "ccc<eol>" --
# \ ud1 -- ud2
#> \ xd -- c-addr u
#S \ ud1 -- ud2
#TIB \ -- n
+ \ n1 n2 -- n1+n2
+! \ n addr --
+DIGIT \ d1 n --d2
+LOOP \ C: do-sys -- ; Run: n -- ; R loop-sys1 -- | loop-sys2
< \ n1 n2 -- t/f
<# \ --
<= \ n1 n2 -- t/f
<> \ n1 n2 -- flag
<BUILDS \ --
<TICKS> \ -- adr
= \ n1 n2 -- flag
> \ n1 n2 -- t/f
>= \ n1 n2 -- t/f
>BODY \ xt -- pfa
>IN \ -- n
>NAME \ cfa -- nfa
>NUMBER \ ud1 c-addr1 u1 --ud2 c-addr2 u2
>R \ x -- ; R -- x
$. \ c-addr --
0< \ n -- flag
0<> \ n -- flag
0= \ n -- flag
0> \ n -- flag
1- \ n -- n-1
1+ \ n -- n+1
2- \ n -- n-2
2! \ d addr --
2@ \ addr -- d
2* \ n1 -- n2
2/ \ n1 -- n2
2+ \ n -- n+2
2DROP \ n1 n2 --
2DUP \ n1 n2 -- n1 n2 n1 n2
2OVER \ n1 n2 n3 n4 -- n1 n2 n3 n4 n1 n2
2SWAP \ n1 n2 n3 n4 -- n3 n4 n1 n2
ABORT" \ Comp "ccc<quote>" -- ; Run: i*x x1 --| i*x ; R j*x --| j*x
ABS \ n1 --|n1|
ACCEPT \ c-addr +n1 --+n2
AGAIN \ C: dest -- ; Run: --
ALIGN \ --
ALIGNED \ addr --addrÕ
ALIGNED \ addr --addr'
ALLOT \ n --
AND \ n1 n2 -- n3
BEGIN \ C: -- dest ; Run: --
BL \ -- char
BOUNDS \ addr len -- addr+len addr
C, \ char --
C! \ b addr --
C!F \ b addr --
C" \ Comp "ccc<quote>" -- ; Run: -- c-addr
C@ \ addr -- b
CATCH \ i*x xt -- j*x 0|i*x n
CELL \ -- 2
CMOVE \ source dest len --
CMOVE> \ source dest len --
COLD \ --
COMMIT \ xt|0 --
COMPILE, \ addr --
CONSOLE \ -- addr
CONSOLE0 \ -- addr
CONSTANT \ x "<spaces>name" -- ; Exec --x
COUNT \ addr --addr+1 len
CR \ --
CREATE \ --
D- \ d1 d2 -- d1-d2
D. \ d --
D.R \ dn --
D+ \ d1 d2 -- d3
D< \ d1 d2 -- t/f
D= \ d1 d2 -- t/f
D> \ d1 d2-- t/f
D0= \ d -- t/f
DABS \ d1 --|d1|
DECIMAL \ --
DEFER \ Comp "<spaces>name" -- ; Run: i*x --j*x
DEPTH \ ??? -- +n
DIGIT \ char base -- 0 | n true
DISK-ERROR \ -- addr ; variable
DNEGATE \ d1 ---d1
DO \ C: -- do-sys ; Run: n1|u1 n2|u2 -- ; R -- loop-sys
DOES> \ C: colon-sys1 -- colon-sys2 ; Run: -- ; R nest-sys --
DP \ -- addr
DPL \ -- addr
DROP \ n1 --
DUMP \ addr len --
DUP \ n1 -- n1 n1
ELSE \ C: orig1 -- orig2 ; Run: --
EMIT \ --char
EMPTY \ --
END-LOAD \ --
EQU \ x "<spaces>name" -- ; Exec --x
ERASE \ addr len --
EVALUATE \ i*x c-addr u -- j*x
EXECUTE \ xt --
EXIT \ R nest-sys --
FENCE \ -- addr
FILL \ addr len char --
FIND \ c-addr -- c-addr 0|xt 1|xt -1
FLERASE \ addr len --
GREEN-OFF \ -- ; P1.6
GREEN-ON \ -- ; P1.6
HERE \ --addr
HEX \ --
HOLD \ char --
I \ --n
IF \ C: -- orig ; Run: x--
IMMEDIATE \ --
INCLUDE \ "<filename>" --
INTERPRET \ --
INVERT \ n1 --n2
IPVEC \ -- addr
IS \ "<spaces>name" --
J \ --n
KEY \ --char
KEY? \ --flag
LATER \ n --n'
LEAVE \ --
LEDACTIVE \ -- addr
LITERAL \ Comp x -- ; Run: -- x
LOOP \ C: do-sys -- ; Run: -- ; R loop-sys1 -- | loop-sys2
LSHIFT \ x count --xÕ
M* \ n1 n2 -- d
MAX \ n1 n2 -- max(n1,n2)
MIN \ n1 n2 -- min(n1,n2)
MOD \ n1 n2 -- rem
MOVE \ addr1 addr2 u --
MS \ n --
MU/MOD \ ud1 u2 --u3 ud4
NAME> \ nfa -- cfa
NEGATE  \ n1 ---n1
NIP \ n1 n2 -- n2
NOOP \ -- ; dummy
NUMBER? \ $addr -- n1|d2|0
OFF \ addr --
ON \ addr --
OPVEC \ -- addr
OR \ n1 n2 -- n3
```

MSP430 Lite Target Glossary – User WORDS of stand alone version - MCU is MSP430G2553

ORG \ addr --
OUT \ -- n
OVER \ n1 n2 -- n1 n2 n1
PAD \ -- addr
PARSE \ char "ccc<char>" --c-addr u
PAUSE \ --
PICK \ nn..n0 n -- nn..n0 nn
PLACE \ c-addr1 u c-addr2 --
POSTPONE \ Comp "<spaces>name" --
QUERY \ --
QUIT \ -- ; R i*x --
R@ \ --x ; R x -- x
R> \ --x ; R:x--
R0 \ -- addr
RALLOT \ n --
RAM \ --
REBOOT \ --
RECURSE \ Comp --
RED-OFF \ -- ; P1.0
RED-ON \ -- ; P1.0
REPEAT \ C: orig dest -- ; Run: --
RHERE \ --addr
ROM \ --
ROT \ n1 n2 n3 -- n2 n3 n1
RP \ -- addr
RSHIFT \ x count --xÕ
S" \ Comp "ccc<quote>" -- ; Run: -- c-addr u
S= \ addr1 addr2 count --flag
S>D \ n -- d
S0 \ -- addr
SCAN \ caddr u char -- caddr2 u2
SEARCH-WORDLIST \ c-addr u wid --0|xt 1|xt -1
SELF \ task identifier and TCB
SKIP \ c-addr u char -- 'c-addr 'u
SM/REM \ d n --rem quot
SPACE \ --
SPACES \ n --
START-CLOCK \ --
STOP-CLOCK \ --
SWAP \ n1 n2 – n2 n1
TESTAPP \ -- ; enless loop, reset to leave loop.
THEN \ C: orig -- ; Run: --
THROW \ k*x n -- k*x|i*x n
TICKS \ -- n
TICKS \ -- n
TIMEDOUT? \ n -- flag
TUCK \ n1 n2 -- n2 n1 n2
TYPE \ c-addr len --
TYPE \ c-addr len --
U. \ u --
U< \ n1 n2 -- flag
U> \ n1 n2 -- flag
U2/ \ n1 -- n2
UM* \ u1 u2 -- ud
UM/MOD \ u32 u16 --urem uquot
UNLOOP \ --
UNTIL \ C: dest – ; Run: x --
UPC \ char -- charÕ
UPPER \ c-addr len --
USER \ u "<spaces>name" -- ; Exec --addr
VARIABLE \ "<spaces>name" -- ; Exec --a-addr
WHILE \ C: dest -- orig dest ; Run: x --
WITHIN \ n1|u1 n2|u2 n3|u3 -- flag
WORD \ char "<chars>ccc<char>" -- c-addr
WORDS \ --
XDP \ --addr
XOR \ n1 n2 -- n3