Micromite MMBasic
Version 5.4
Quick Reference

Program Management

```
CONTINUE
CPU speed
CPU SLEEP [ sec [, abortpin]]
CPU RESTART
CSUB name type [, type ... ]
END CSUB
CFUNCTION name( type [,type ... ] )  typer
END CFUNCTION
DEFINEFONT #Nbr
END DEFINEFONT
EDIT
END
LIBRARY SAVE | DELETE | LIST
LIST  [ALL]
MEMORY
NEW
nbr = PEEK(BYTE | WORD | VARADDR | CFUNADDR
              | VARTBL, |  PROGMEM,  | VAR args)
POKE BYTE | WORD | VARTBL, | VAR addr, dat
RUN
TIMER = msec
TRACE ON | OFF | LIST nn
VAR SAVE var [, var]...  | RESTORE | CLEAR
WATCHDOG timeout | OFF
XMODEM SEND | RECEIVE  [filename$]
```

Input/Output

```
SETPIN pin, cfg [, option]
cfg = OFF | AIN | DIN | FIN | PIN | CIN | DOUT
option = PULLUP | PULLDOWN | OC | gate | cycles
SETPIN pin, OFF | INTH | INTL | INTB, target [, option]
option = PULLUP | PULLDOWN
PIN( pin ) = value
PORT(start, nbr [,start, nbr]...) = value
PULSE pin, width
pulsewidth = PULSIN( pin, polarity [, t1 [, t2]])
value = PIN(pin)
value = PORT(start, nbr [,start, nbr]...)
```

Commands

```
' (single quotation mark) - comment
? (question mark) – shorthand for PRINT
CLEAR
CONST id1 = expression [,  id2 = expression, ...]
CONTINUE DO | FOR
DATA constant  [, constant, ...]
DATE$ = "DD-MM-YY" | "DD/MM/YY"
DIM  [AS] [type] var [, var, ...] [AS type [, var AS type , ...]]
DO [WHILE <test>]
LOOP
DO
LOOP UNTIL <test>
ERASE array [, array, ... ]
ERROR [message$]
EXIT DO | FOR | FUNCTION | SUB
FOR var = start TO finish [STEP increment]
NEXT [var1 [, var2, ...]
FUNCTION name( [arg1 [AS type] [,arg2, ...]] ) [AS type]
END FUNCTION
GOSUB target
RETURN
GOTO target
IF <test> THEN <stmt> [ELSE <stmt>] [: <stmt> : ... ]
IF <test> THEN — ELSEIF — ELSE — ENDIF
INPUT ["prompt string$" ;|, ]  var [, var, ...]
LINE INPUT ["prompt string$",]  var$
LET variable = expression
variable = expression
LOCAL [type] decl [, decl, ...] [AS type [, var AS type , ...]]
ON ERROR ABORT | IGNORE | SKIP [nn] | CLEAR
ON nbr GOTO | GOSUB target1 [, target2, ...]
ON KEY subroutine
PAUSE ms
PRINT expression1 [, | ;]  [expression2, ...]   [, | ;]
RANDOMIZE nbr
READ var1[, var2, ...]
RESTORE [line]
REM comment
SELECT CASE  —   CASE [ELSE]  —  END SELECT
SETTICK period, target [, nbr]
SUB name arg1 [AS type] [, arg2 [AS type], ... ]
END SUB
TIME$ = "HH:MM:SS" | "HH:MM" | "HH"
```

Functions

```
ACOS( nbr )
ASIN( nbr )
COS( nbr )
EXP( nbr )
PI
SIN( nbr )
TAN( nbr )
ABS( nbr )
ATN( nbr )
DEG( radians )
LOG( nbr )
RAD( degrees )
SQR( nbr )
EVAL( str$ )

CINT( nbr )
INT( nbr )
FIX( nbr )

ASC( str$ )
CHR$( nbr )
INSTR([start,] str$, pat$)
LEFT$( str$, nbr )
LEN( str$ )
OCT$( nbr [, chars])
STR$( nbr [, m [, n [, c$]]] )
STRING$( nbr,  ascii  | str$ )
LCASE$( str$ )
VAL( str$ )
BIN$( nbr  [, chars])
HEX$( nbr [, chars])
RIGHT$( str$, nbr )
MID$( str$, start [, nbr])
SPACE$( nbr )
UCASE$( str$ )

DATE$
TIMER
MAX( nbr [, nbr [, ..]] )
POS
SGN( nbr )
TIME$
INKEY$
MIN( nbr [, nbr [, ...]] )
RND( nbr )
TAB( nbr )
```

Options

```
OPTION AUTORUN OFF | ON
OPTION BASE 0 |  1
OPTION BAUDRATE nbr
OPTION BREAK nn
OPTION CASE UPPER | LOWER | TITLE
OPTION CLOCKTRIM ±n
OPTION COLOURCODE ON | OFF
OPTION CONSOLE ECHO | NOECHO
OPTION CONSOLE INVERT | NOINVERT
OPTION CONSOLE AUTO
OPTION DEFAULT FLOAT | INTEGER |  STRING | NONE
OPTION DISPLAY lines [,chars]
OPTION ERROR CONTINUE | ABORT
OPTION EXPLICIT
OPTION KEYBOARD US | UK | FR | GR | BE | IT | ES
OPTION LIST
OPTION PIN nbr
OPTION RESET
OPTION TAB 2 | 4 | 8
```

Operators

NOT ^	Logical inverse, exponentiation
* / \	Multiply, division (float & integer)
MOD	Modulus (remainder)
+ -	Addition and subtraction
x << y x >> y	Shift bits left/right by y bits
= <> < >	Equals, not equals, less/greater than
<= >=	Less/greater than or equals
AND OR XOR	Logical and, or, exclusive or

Variables

Identifier = [A-Z | _] [A-Z | 0-9 | . | _]   Max 32 chars.
Variable Suffix: FLOAT = ! INTEGER = % STRING = $
Number Prefix: [&H | &O | &B] number

MM.VER
MM.ERRNO
MM.HRES
MM.FONTHEIGHT
MM.DEVICE$
MM.ERRMSG$
MM.VRES
MM.FONTWIDTH

MM.WATCHDOG
MM.I2C
MM.ONEWIRE

GUI Controls (MM+)

OPTION CONTROLS nn

GUI AREA #ref, X, Y [, width, height]
GUI BUTTON #ref, caption$, X, Y [, w, h, FC, BC]
GUI CAPTION #ref, text$, X, Y [, just$, FC], BC]
GUI CHECKBOX #ref, caption$, X, Y [, size, colour]
GUI DISPLAYBOX #ref, X, Y [, width, height, FC, BC]
GUI FRAME #ref, caption$, X, Y [, width, height, colour]
GUI LED #ref, caption$, X, Y [, radius, colour]
GUI NUMBERBOX #ref, X, Y [, width, height, FC, BC]
GUI RADIO #ref, caption$, X, Y [, radius, colour]
GUI SPINBOX #ref, X, Y, w, h [, FC, BC, Step, Min, Max]
GUI SWITCH #ref, caption$, X, Y [, width, height, FC, BC]
GUI TEXTBOX #ref, X, Y [, width, height, FC, BC]

GUI DELETE #ref1 [,#ref2, ...] | ALL
GUI DISABLE #ref1 [,#ref2, ...] | ALL
GUI ENABLE #ref1 [,#ref2, ...] | ALL
GUI HIDE #ref1 [,#ref2, ...] | ALL
GUI REDRAW #ref1 [,#ref2, ...] | ALL
GUI SHOW #ref1 [,#ref2, ...] | ALL

GUI NUMBERBOX CANCEL
GUI TEXTBOX CANCEL

GUI BCOLOUR colour, #ref1 [, #ref2, ...]
GUI FCOLOUR colour, #ref1 [, #ref2, ...]
GUI BEEP msec

GUI INTERRUPT down [, up]
coordinate = TOUCH(DOWN | UP | LASTX | LAST)
control = TOUCH(REF | LASTREF)
value = CTRLVAL(#ref)   CTRLVAL(#ref) = value
GUI SETUP #n   PAGE #n [,#n2, ...]
button = MSGBOX (msg$, b1$ [,b2$ [, b3$ [, b4$]]])

Communications & File I/O

OPEN C$ AS #fnbr
C$ = "COMn: baud, buf, int, nbr, DE, 9BIT, INV, OC, S2"

I2C OPEN speed, timeout [, PU]
I2C WRITE addr, option, sendlen, data [,data ....]
I2C READ addr, option, rcvlen, rcvbuf
I2C SLAVE OPEN addr, mask, opt, i_send, i_rcv
I2C SLAVE WRITE len, data [, data ....]
I2C SLAVE READ len, buf, rcvd
I2C [SLAVE] CLOSE

ONEWIRE READ pin, flag, len, data, ...
ONEWIRE WRITE pin, flag, len, data, ...
ONEWIRE RESET pin

SPI[2] OPEN speed, mode, bits
received_data = SPI[2](data_to_send)
SPI[2] WRITE nbr, data1,, ... | str$ | array()
SPI[2] READ nbr, array()
SPI[2] CLOSE

OPTION SDCARD CS [, CD [,WP]] | DISABLE
OPEN fname$ FOR mode AS [#]fnbr
‘mode’ = INPUT | OUTPUT | APPEND | RANDOM

LOAD file$ [,R]   LOAD IMAGE file$ [, x, y]
MKDIR dir$ RMDIR dir$
CHDIR dir$ dir = CWD$
NAME old$ AS new$ KILL file$
SAVE [file$]   SAVE IMAGE file$
SEEK [#]fnbr, pos   FILES [fspec$]
fname$ = DIR$([fspec [, type]])

CLOSE [#]fnbr [,[#]fnbr] ...
State = EOF([#]fnbr)
INPUT #fnbr, var1 [, var2, ...]
LINE INPUT #fnbr, string variable$
PRINT #fnbr, expression1 [, | ;] [expression2, ...] [, | ;]
INPUT$(nbr, [#]fnbr)
nbr = LOC([#]fnbr)   nbr = LOF([#]fnbr)

PLAY TONE left [, right [, duration]]
PLAY WAV file$ [, interrupt]
PLAY PAUSE | RESUME | STOP | VOLUME left, right

Micromite MMBasic V5.4

(Micromite Plus extra features are in red)

Downloads: http://geoffg.net/micromite.html
Forum: http://www.thebackshed.com/forum/Microcontrollers

Devices

IR dev, key , int | CLOSE
KEYPAD var, int, r1, r2, r3, r4, c1, c2, c3 , c4 | CLOSE
LCD INIT d4, d5, d6, d7, rs, en
LCD line, pos, text$ | CLEAR | CLOSE
LCD CMD | DATA d1 [, d2 [, etc]]
PWM channel, freq, pwm1 [, pwm2 [, pwm3]]
PWM channel, STOP
RTC GETTIME
RTC SETTIME year, month, day, hour, minute, second
RTC SETREG | GETREG register, value | var
OPTION RTC data, clock | DISABLE
SERVO channel [, freq], out1 [, out2 [, out3]]
SERVO channel, STOP
TEMPR START pin [, precision 0 to 3]
Temperature = TEMPR(pin)

LCD Display Panel

OPTION LCDPANEL type, orient, D/C, reset [,CS]
type = ILI9163 | ST7735 | ILI9341

OPTION LCDPANEL type, orient [, LCD-A] [, readpin]
type = SSD1963_[4][5][5A][7][7A][8]

OPTION LCDPANEL CONSOLE [font [, fc [, bc , [blight]]]]
OPTION LCDPANEL NOCONSOLE

OPTION LCDPANEL DISABLE
GUI CALIBRATE
GUI RESET LCDPANEL
GUI TEST LCDPANEL | TOUCH

OPTION TOUCH T_CS pin, T_IRQ pin [, click pin]
OPTION TOUCH DISABLE

PIXEL x, y [, colour]
LINE x1, y1, x2, y2 [, lw [, colour]]
CIRCLE x, y, r [, lw] [, a] [, colour] [, fill]
TRIANGLE x1, y1, x2, y2, x3, y3 [, colour [, fill]]
BOX x1, y1, w, h [, lw] [, colour] [, fill]
RBOX x1, y1, w, h [, rc] [, colour] [, fill]
TEXT x, y, str$ [, just$] [, fnt] [, scale] [, colour] [, bc]
GUI BITMAP x, y, data [, w] [, h] [, s] [, colour] [, bc]

CLS [colour]
COLOUR fore [, back]
COLOR fore [, back]
FONT [#]font-number, scaling
BACKLIGHT percent
BLIT READ | WRITE [#]buffer, x, y, w, h
BLIT CLOSE [#]buffer
BLIT x1, y1, x2, y2, w, h

colour% = RGB(red, green, blue | colour listed below)
white black blue green cyan red magenta yellow brown gray

coordinate = TOUCH(X | Y)