```
================================================================================
Projekt : STM32F429-Basic
Version : C.01
Datum   : 02.11.2014
EMail   : mc-4u(@)t-online.de
Web     : www.mikrocontroller-4u.de
================================================================================
```

Inhalt

Seite 2 = Syntaxhinweise / Konventionen
Seite 3 = Rechenoperationen   [*,-,*,/,%]
Seite 4 = Logik-Operationen   [&,|,^]
Seite 5 = Grundbefehle   [REM, LET, PRINT, IF/THEN/ELSE]
                                [FOR/TO/STEP, NEXT, GOTO, GOSUB, RETURN, END]
Seite 7 = System-Befehle   [CONST, CMD, RND, PEEK, POKE]
                                [PAUSE, CLRTIC, GETTIC]
Seite 9 = LCD-Befehle   [CHR, HEX, CLS, CURSOR, INK]
                                [PAPER, SETPX, GETPX, LINE]
                                [RECT, CIRCLE]
Seite 12 = GPIO-Befehle   [INITPIN, IN, OUT, ADC]
Seite 13 = Tastatur-Befehle   [GETKEY]

Syntax Hinweise

```
-------------------------------------------------------------------------
<var>   = Variablen-Name            : 'a' bis 'z'
<uint>  = unsigned Zahlenwert       : '0' bis '99999'
<int>   = signed Zahlenwert         : '-99999' bis '99999'
<hex>   = Hex Wert                  : '$"0"' bis '$"FFFF"'
<const> = Konstante                 : '#"COL_RED"'
<str>   = String                    : '"Hallo"'
<tr>    = Trennungszeichen          : ';' oder ','
<rel>   = Relation                  : '=' oder '<' oder '>'
<cmd>   = beliebiger Basic-Befehl   : 'PRINT 12'
-------------------------------------------------------------------------
<exp>   = Ausdruck vom Typ          : <var>,<int>,<hex>,<const>
<sexp>  = Ausdruck vom Typ          : <var>,<int>,<hex>,<const>,<str>
-------------------------------------------------------------------------
Klammersetzung um <exp>            : (a+3)*2
-------------------------------------------------------------------------
```

Konventionen

1. Jede Basic-Zeile muss mit einer Zeilen-Nummer <uint> beginnen.
2. Es darf keine Zeilen-Nummer doppelt geben.
3. Die Zeilen-Nummern müssen aufsteigend sortiert sein.
4. In jeder Basic-Zeile darf nur EIN einzelner Basic-Befehl stehen.
   (ausnahme = IF)
5. Float Zahlen sind nicht zulässig.
6. Ziel von GOTO und GOSUB muss <uint> sein und die Zeilennummer
   muss existieren.
7. Konstanten müssen vor Benutzung per "CONST" initialisiert werden.

```
============================
Rechenoperationen
Ergebnis ist wieder ein <exp>
============================
------------------------------------------------------------------------
Addition :
------------------------------------------------------------------------

Syntax   : <exp>+<exp>
Beispiel : 3+7

------------------------------------------------------------------------
Subtraktion :
------------------------------------------------------------------------

Syntax   : <exp>-<exp>
Beispiel : 3-7

------------------------------------------------------------------------
Multiplikation :
------------------------------------------------------------------------

Syntax   : <exp>*<exp>
Beispiel : 3*7

------------------------------------------------------------------------
Division :
------------------------------------------------------------------------

Syntax   : <exp>/<exp>
Beispiel : 7/3

------------------------------------------------------------------------
Modulo :
------------------------------------------------------------------------

Syntax   : <exp>%<exp>
Beispiel : 7%3
```

```
============================
Logik-Operationen
Ergebnis ist wieder ein <exp>
============================
------------------------------------------------------------------------
Logisches UND :
------------------------------------------------------------------------

Syntax   : <exp>&<exp>
Beispiel : 3&7

------------------------------------------------------------------------
Logisches ODER :
------------------------------------------------------------------------

Syntax   : <exp>|<exp>
Beispiel : 3|7

------------------------------------------------------------------------
Logisches XOR :
------------------------------------------------------------------------

Syntax   : <exp>^<exp>
Beispiel : 3^7
```

```
============================
Grundbefehle
============================
------------------------------------------------------------------------
REM
Kommentar ohne Funktion
------------------------------------------------------------------------

Syntax   : REM<anything>
Beispiel : REM Test-"1234"

------------------------------------------------------------------------
LET
Wertzuweisung an eine Variable
(LET ist optional)
------------------------------------------------------------------------

Syntax   : [LET]<var>=<exp>
Beispiel : LET a=3+5-b
Beispiel : a=3+5-b

------------------------------------------------------------------------
PRINT
Ausgabe an LCD an der aktuellen Cursor-Position (wenn CMD=0)
oder ueber UART (wenn CMD=1)
------------------------------------------------------------------------

Syntax   : PRINT<sexp>[<tr><sexp><tr><sexp>...]
<tr>     : ';' = unterdrückt Zeilenumbruch
<tr>     : ',' = erzeugt Tabulatorsprung
Beispiel : PRINT a
Beispiel : PRINT "Wert=";a;"mV"
Beispiel : PRINT "X1=";a+b,"X2=";c*d,"X3=";e

------------------------------------------------------------------------
IF/THEN/ELSE
Bedingung
(THEN und ELSE sind optional)
------------------------------------------------------------------------

Syntax   : IF<exp><rel><exp>[THEN]<cmd>[ELSE<cmd>]
Beispiel : IF a>5 THEN b=3 ELSE b=6
Beispiel : IF a>5 b=c+8
Beispiel : IF a>5 b=9 ELSE PRINT "Test"
```

FOR/TO/STEP
Schleifen-Kopf
(STEP ist optional)

Syntax : **FOR<var>=<exp>TO<exp>[STEP<exp>]**
Beispiel : FOR a=2 TO 10 STEP 3
Beispiel : FOR a=b TO 9
Beispiel : FOR a=8 TO 2 STEP c

NEXT
Schleifen-ENDE

Syntax : **NEXT<var>**
Beispiel : NEXT a

GOTO n
Sprung zu Zeilen-Nummer [n]

Syntax : **GOTO<uint>**
Beispiel : GOTO 100

GOSUB n
Sprung zu Zeilen-Nummer [n] (mit Rücksprung bei RETURN)

Syntax : **GOSUB<uint>**
Beispiel : GOSUB 100

RETURN
Ruecksprung nach GOSUB

Syntax : **RETURN**
Beispiel : RETURN

END
beendet das Basic-Programm

Syntax : **END**
Beispiel : END

```
============================
System-Befehle
============================
------------------------------------------------------------------------
CONST "TEXT"=n
Konstante "TEXT" definieren und den Wert [n] zuweisen
------------------------------------------------------------------------

Syntax   : CONST<const>=<exp>
Beispiel : CONST #"COL_RED"=$"F800"
Beispiel : CONST #"MAXX"=320

------------------------------------------------------------------------
CMD n
ChangeMainDevice [n=Ziel] (n:0=LCD, 1=UART)
------------------------------------------------------------------------

Syntax   : CMD<exp>
Beispiel : CMD 1
Beispiel : CMD a

------------------------------------------------------------------------
RND n
Zufallszahl [n] (n=0 bis 999)
------------------------------------------------------------------------

Syntax   : RND<var>
Beispiel : RND a

------------------------------------------------------------------------
PEEK d=(adr)
Auslesen von einem 32bit-Wert [d] aus Adresse [adr] vom Basic-RAM
(adr=0 bis 0x7FFF)
------------------------------------------------------------------------

Syntax   : PEEK<var>=(<exp>)
Beispiel : PEEK a=(1000)
Beispiel : PEEK a=(b)

------------------------------------------------------------------------
POKE adr,d
Schreiben von einem 32bit-Wert [d] in Adress [adr] vom Basic-RAM
(adr=0 bis 0x7FFF)
(d=-99999 bis 99999)
------------------------------------------------------------------------

Syntax   : POKE<exp>,<exp>
Beispiel : POKE 1000,-96
Beispiel : POKE 1000,a
```

PAUSE n
Pause von [n] ms (n>=0)

Syntax : **PAUSE<exp>**
Beispiel : PAUSE 100
Beispiel : PAUSE a

CLRTIC
löschen vom Basic-Timer

Syntax : **CLRTIC**
Beispiel : CLRTIC

GETTIC n
auslesen vom Basic-Timer [n] ms

Syntax : **GETTIC<var>**
Beispiel : GETTIC a

```
============================
LCD-Befehle
============================
-------------------------------------------------------------------------
CHR n
Anzeige vom Ascii-Zeichen mit der Nummer [n] (n=0 bis 255)
-------------------------------------------------------------------------

Syntax   : CHR<exp>
Beispiel : CHR 65
Beispiel : CHR a

-------------------------------------------------------------------------
HEX n[,z]
Anzeige der Zahl [n] in HEX-Schreibweise mit [z] Ziffern
(z=optional)
-------------------------------------------------------------------------

Syntax   : HEX<exp>[,<exp>]
Beispiel : HEX 12
Beispiel : HEX a,4

-------------------------------------------------------------------------
CLS
löschen vom Bildschirm (mit Hintergrundfarbe)
-------------------------------------------------------------------------

Syntax   : CLS
Beispiel : CLS

-------------------------------------------------------------------------
CURSOR x,y
positionieren vom Cursor für Textausgabe
(x=0 bis 319, y=0 bis 239)
-------------------------------------------------------------------------

Syntax   : CURSOR<exp>,<exp>
Beispiel : CURSOR 289,26
Beispiel : CURSOR a,b

-------------------------------------------------------------------------
INK c
setzen der Vordergrundfarbe [c] R5G6B5 (c=0 bis 65535)
-------------------------------------------------------------------------

Syntax   : INK<exp>
Beispiel : INK $"F800"
Beispiel : INK a
```

PAPER c
setzen der Hintergrundfarbe [c] R5G6B5 (c=0 bis 65535)

Syntax : **PAPER<exp>**
Beispiel : PAPER $"F800"
Beispiel : PAPER a

SETPX x,y[,c]
zeichnet ein Pixel an x,y mit Farbe [c]
(x=0 bis 319, y=0 bis 239)
[c] R5G6B5 (c=0 bis 65535)
(c=optional)

Syntax : **SETPX<exp>,<exp>[,<exp>]**
Beispiel : SETPX 123,45,$"F800"
Beispiel : SETPX a,b

GETPX c=(x,y)
auslesen der Farbe [c] vom Pixel x,y
(x=0 bis 319, y=0 bis 239)
[c] R5G6B5 (c=0 bis 65535)

Syntax : **GETPX<var>=(<exp>,<exp>)**
Beispiel : GETPX c=(123,45)
Beispiel : GETPX c=(a,b)

LINE x1,y1,x2,y2[,c]
zeichnet Linie von x1,y1 nach x2,y2 mit Farbe [c]
(x=0 bis 319, y=0 bis 239)
[c] R5G6B5 (c=0 bis 65535)
(c=optional)

Syntax : **LINE<exp>,<exp>,<exp>,<exp>[,<exp>]**
Beispiel : LINE 10,20,100,120,$"F800"
Beispiel : LINE a,b,c,d

```
------------------------------------------------------------------------
RECT x1,y1,x2,y2,m[,c]
zeichnet ein Rechteck von x1,y1 nach x2,y2 mit Farbe [c] und Mode [m]
(x=0 bis 319, y=0 bis 239)
(m:0=outline, 1=filled)
[c] R5G6B5 (c=0 bis 65535)
(c=optional)
------------------------------------------------------------------------

Syntax   : RECT<exp>,<exp>,<exp>,<exp>,<exp>[,<exp>]
Beispiel : RECT 10,20,100,120,0,$"F800"
Beispiel : RECT a,b,c,d,1

------------------------------------------------------------------------
CIRCLE x,y,r,m[,c]
zeichnet ein Kreis an x,y mit Radius [r] mit Farbe [c] und Mode [m]
(x=0 bis 319, y=0 bis 239)
(m:0=outline, 1=filled)
[c] R5G6B5 (c=0 bis 65535)
(c=optional)
------------------------------------------------------------------------

Syntax   : CIRCLE<exp>,<exp>,<exp>,<exp>[,<exp>]
Beispiel : CIRCLE 100,120,30,0,$"F800"
Beispiel : CIRCLE a,b,c,1
```

```
============================
GPIO-Befehle
============================
-------------------------------------------------------------------------
INITPIN pin,typ[,mode]
initialisiert den GPIO-Pin [pin] mit einem [typ] und [mode]
pin : 0x00=PA0, 0x01=PA1...0x0F=PA15
      0x10=PB0, 0x11=PB1...0x1F=PB15
      0x60=PG0, 0x61=PG1...0x6F=PG15
typ : 0=Digital-Eingang -> mode:0=NoPull, 1=PullUp, 2=PullDown
    : 1=Digital-Ausgang -> mode:0=Lo-Pegel, 1=Hi-Pegel
    : 2=Analog-Eingang  -> mode:nc
(mode=optional)
-------------------------------------------------------------------------

Syntax   : INITPIN<exp>,<exp>[,<exp>]
Beispiel : INITPIN $"6D",1
Beispiel : INITPIN $"00",0

-------------------------------------------------------------------------
IN v=(pin)
einlesen vom Pegel eines Digital-Eingangs
pin : erlaubt = PA0,5,7,9,10 / PB2,4,7 / PC3,8,11,12,13,14,15
                PD2,4,5,7 / PE2,3,4,5,6 / PF6 / PG2,3,8,13,14
v   : 0=Lo-Pegel, 1=Hi-Pegel
-------------------------------------------------------------------------

Syntax   : IN<var>=(<exp>)
Beispiel : IN a=($"00");

-------------------------------------------------------------------------
OUT pin,v
ausgeben eines Werts an einen Digital-Ausgang
pin : erlaubt = PA0,5,7,9,10 / PB2,4,7 / PC3,8,11,12,13,14,15
                PD2,4,5,7 / PE2,3,4,5,6 / PF6 / PG2,3,8,13,14
v   : 0=Lo-Pegel, 1=Hi-Pegel, 2=Toggeln
-------------------------------------------------------------------------

Syntax   : OUT<exp>,<exp>
Beispiel : OUT $"6D",1

-------------------------------------------------------------------------
ADC v=(pin[,mw])
einlesen vom Wert eines Analog-Eingangs
pin : erlaubt = PA5, PA7, PC3
mw  : [0...7] anzahl der Mittelwerte (Anzahl = 2^mw)
(mw=optional)
v   : 0 bis 4095
-------------------------------------------------------------------------

Syntax   : ADC<var>=(<exp>[,<exp>])
Beispiel : ADC a=($"00",4);
```

```
============================
Tastatur-Befehle
============================
-------------------------------------------------------------------------
GETKEY k1[,k2,shift]
einlesen vom Tastaturwert [k1] und [k2] und dem Shift-Wert [shift]
(k2,shift=optional)
k1    : Tasten-Code von Taste-1 (0=keine Taste)
k2    : Tasten-Code von Taste-2 (0=keine Taste)
shift : BIT0=LSHIFT, BIT1=RSHIFT, BIT2=STRG, BIT3=ALT, BIT4=ALTGR
-------------------------------------------------------------------------

Syntax   : GETKEY<var>[,<var>,<var>]
Beispiel : GETKEY a,b
```