320x320 LCD Controller

- LCD Controller für ein 320x240 4bit LCD.
- Schnittstelle: UART
- Optimiert für Textausgabe
- vollständiger 8x12 Zeichensatz (256 Zeichen)
- 40x20 Zeichen
- Grafik möglich (Funktionen für Pixel, Linien, Rechtecke, Bilder vorhanden)
- 4 Graustufen (2bpp)

Befehl	Parameter	Beschreibung
0		NOP (nichts machen)
1		Cursor auf (0,0) setzen
2		LCD (Kontrastspannung) aus
3		LCD (Kontrastspannung) an
8		Backspace
9	X Low X High Y Farbe	Set Pixel (x,y,Farbe)
10		Cursor eine Zeile nach unten
11		Zeichen an Cursorposition löschen
12		Komplettes Bild löschen
13		Cursor an Zeilenanfang setzen
14		Nicht verwenden!
15	Kontrast	Kontrast einstellen (0-100)
16	0xAA X Y XS YS Modus Daten…	Bild an Position (x,y) mit Auflösung (xs,ys) laden. Auf diesen Befehl folgen xs*ys (*2) Bytes mit Bilddaten. x und xs zählen Bytes (Nibbels). x hat daher den Bereich 0-79 (0-159), xs den Bereich 1-80 (1-160). y und ys zählen Zeilen. y hat daher den Bereich 0-239, ys den Bereich 1-240. Modus schaltet zwischen 1bpp (0) und 2bpp (1) um. 0xAA dient nur als Sicherheitsbyte, damit nicht versehentlich dieser Befehl (z.B. aufgrund eines Übertragungsfehlers) ausgeführt wird.
17	X Y	Textcursor auf (x,y) setzen X zählt in 8 Pixel Schritten, Y in Zeilenschritten
18	X1 Low X1 High Y1 X2 Low X2 High Y2 Farbe	Linie von (x1,y1) nach (x2,y2) mit Farbe zeichnen
19	X1 Low X1 High Y1 R Farbe 1 Farbe 2	Kreis mit Mittelpunkt (x,y) und Radius R zeichnen. Farbe 1 ist die Füllfarbe. Farbe 2 ist die Rahmenfarbe. Hinweis: Der Radius darf nicht größer sein als x oder y, ebenso darf er nicht 0 sein, ansonsten wird der Befehl ignoriert.
21	Textfarbe Hintergrundfarbe	Setz Textfarbe und Hintergrundfarbe
22		Nicht verwenden!
25	Nummer Zeile 1 Zeile 2 … Zeile 8	Benutzerdefiniertes Zeichen an die entsprechende Position laden. Jedes Zeichen besteht aus 8 Zeilen (Bytes) zu je 8 Pixeln (Bits). Insgesamt sind 16 Zeichen verfügbar (Nummer 0-15)
28		Nicht verwenden!
29	X1 Low X1 High Y1 X2 Low X2 High Y2	Rechteck mit Eckkoordinaten (x1,y1, x2,y2) zeichnen. Farbe 1 ist die Füllfarbe. Farbe 2 ist die Rahmenfarbe.

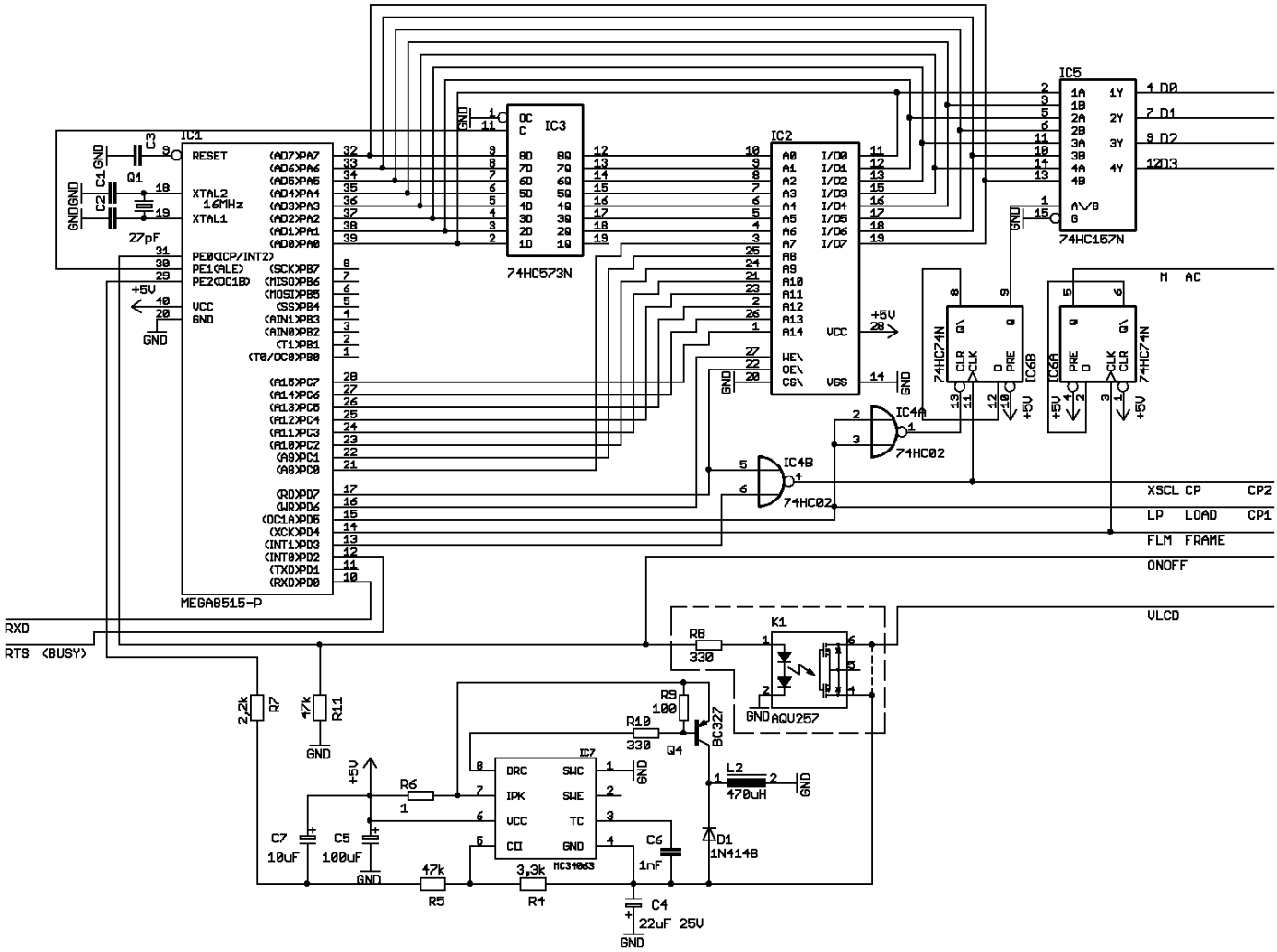
	Farbe 1 Farbe 2	
30	Ziel Wert	Zeichen aus erweitertem Zeichensatz schreiben. Ziel = 0: Benutzerdefiniertes Zeichen 0-15 schreiben Ziel > 0: ASCII Zeichen 0-255 schreiben.
32-255		Buchstaben an Cursorposition zeichnen, Cursor erhöhen.

Ein paar Beispiele:

29 140 0 100 179 0 139 0 255 — zeichnet ein 40x40 großes, schwarzes Rechteck, mit einem weißen Rand

19 160 0 120 15 64 128 — zeichnet einen Kreis mit dem Mittelpunkt 160,120 und dem Radius 15, gefüllt mit der dunkelsten Graustufe und einem etwas helleren Rand

16 170 0 1 8 0 1 2 4 8 16 32 64 128 — zeichnet ein 8x8 Pixel großes Bild mit den Daten 1, 2, 4, 8, 16, 32, 64, 128 (=schräge Linie)



Der OptoMOSFET K1 kann bei LCDs die über einen ONOFF oder DispOFF\ Pin verfügen entfallen. Falls der OptoMOSFET nicht erhältich ist, kann dieser durch einen normalen Optokoppler ersetzt werden, wenn dieser genügend Strom für das LCD schalten kann.

R5/R4 stellen die Spannung für das Display und somit den Kontrast ein. Der Kontrast lässt sich über einen kleinen Bereich per Software über PWM regeln. Sollte dieser Bereich für das LCD nicht passen, müssen R5/R4 angepasst werden.

Hinweis zur Ansteuerung: Aufgrund der teilweise ziemlich rechenintensiven Grafikbefehle sollte der RTS/Busy Ausgang vor dem Senden eines Befehls abgefragt werden. Der Controller verfügt zwar über einen 256 Byte Befehlspuffer, aber auch dieser ist irgendwann voll, wenn mehrere aufwendige Befehle ohne Pause gesendet werden! Solange der RTS/Busy Pin Low ist, kann man gefahrlos mindestens 64 Bytes senden, da RTS/Busy aktiviert wird, wenn der Puffer zu ¾ gefüllt ist. Man braucht also nicht vor jedem Byte RTS/Busy zu prüfen, sondern es reicht einmal vor jedem Befehl (falls dieser kürzer als 64 Bytes ist.)

Falls zusätzliche Grafikfunktionen in die Software eingebaut werden sollten, hier eine kurze Beschreibung der Schnittstelle:

```
void lcd_writecgram(    unsigned char x,            // X Position in 8 Pixel Schritten, Bereich 0-79
                        unsigned char y,            // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char nr,           // Zeichennummer, Bereich 0-15
                        unsigned char textcol,      // Textfarbe (2bpp, an MSB ausgerichtet)
                        unsigned char backcol);     // Hintergrundfarbe (2bpp, an MSB ausgerichtet)
```
Schreibt ein benutzerdefiniertes Zeichen auf das LCD an die angegebene Position (x,y).

```
void lcd_writechar (    unsigned char x,            // X Position in 8 Pixel Schritten, Bereich 0-79
                        unsigned char y,            // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char c,            // Zeichen, Bereich 0-255
                        unsigned char tcol,         // Textfarbe (2bpp, an MSB ausgerichtet)
                        unsigned char bcol);        // Hintergrundfarbe (2bpp, an MSB ausgerichtet)
```
Schreibt einen 8x12 Buchstaben auf das LCD an die angegebene Position (x,y).

```
void lcd_writebyte (    unsigned char x,            // X Position in 8 Pixel Schritten, Bereich 0-79
                        unsigned char y,            // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char c);           // Datenbyte das 8 Pixel enthält
```
Schreibt 1 Byte Pixeldaten (8bit) auf das LCD an die angegebene Position (x,y).

```
void lcd_writebyteg (   unsigned char x,            // X Position in 4 Pixel Schritten, Bereich 0-159
                        unsigned char y,            // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char c);           // Datenbyte das 4 Pixel enthält
```
Schreibt 1 Byte Pixeldaten (4 Pixel) mit 2bpp Farben auf das LCD an die angegebene Position (x,y).

```
void lcd_writebyteex (  unsigned char x,            // X Position in 8 Pixel Schritten, Bereich 0-79
                        unsigned char y,            // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char c,            // Datenbyte das 8 Pixel enthält
                        unsigned char textcol,      // Farbe die den 1en aus dem Byte zugewiesen
                        unsigned char backcol);     // Farbe die den 0en aus dem Byte zugewiesen
```
Schreibt 1 Byte Pixeldaten (8bit) auf das LCD und erweitert die Farben auf die angegeben Werte.

```
void lcd_setpixel (     unsigned short x,           // X Position in 1 Pixel Schritten, Bereich 0-319
                        unsigned short y,           // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char c);           // Pixelfarbe (2bpp, an MSB ausgerichtet)
```
Zeichnet einen Pixel an (x,y) mit einer bestimmten Farbe.

```
void lcd_string(        unsigned char x,            // X Position in 8 Pixel Schritten, Bereich 0-79
                        unsigned char y,            // Y Position in 1 Pixel Schritten, Bereich 0-239
                        char *txt,                  // Nullterminierter Textstring im Flash
                        unsigned char tcol,         // Textfarbe (2bpp, an MSB ausgerichtet)
                        unsigned char bcol);        // Hintergrundfarbe (2bpp, an MSB ausgerichtet)
```
Schreibt einen Text an Position (x,y) auf das LCD.

```
void lcd_line (         unsigned short x1,          // X Position in 1 Pixel Schritten, Bereich 0-319
                        unsigned short y1,          // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned short x2,          // X Position in 1 Pixel Schritten, Bereich 0-319
                        unsigned short y2,          // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char color);       // Linienfarbe (2bpp, an MSB ausgerichtet)
```
Zeichnet eine Linie von (x1,y1) nach (x2,y2) mit einer bestimmten Farbe.

```
void lcd_block (        unsigned short x1,          // X Position in 1 Pixel Schritten, Bereich 0-319
                        unsigned short y1,          // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned short x2,          // X Position in 1 Pixel Schritten, Bereich 0-319
                        unsigned short y2,          // Y Position in 1 Pixel Schritten, Bereich 0-239
                        unsigned char color,        // Füllfarbe (2bpp, an MSB ausgerichtet)
                        unsigned char color2);      // Rahmenfarbe (2bpp, an MSB ausgerichtet)
```
Zeichnet ein Rechteck auf das LCD und löscht alles was vorher an dieser Stelle war.

```
void lcd_circle(        unsigned short xm,          // X Position Mittelpunkt in 1 Pixel Schritten, Bereich 0-319
                        unsigned short ym,          // Y Position Mittelpunkt in 1 Pixel Schritten, Bereich 0-239
                        unsigned char r,            // Radius in 1 Pixel Schritten, Bereich 1-119
                        unsigned char color1,       // Füllfarbe (2bpp, an MSB ausgerichtet)
                        unsigned char color2);      // Rahmenfarbe (2bpp, an MSB ausgerichtet)
```
Zeichnet ein Kreis auf das LCD und löscht alles was vorher an dieser Stelle war.

```
void lcd_clear(void);
```
Löscht das LCD (setzt alle Pixel auf 0)

```
void lcd_init(void);
```
Startet den LCD Controller