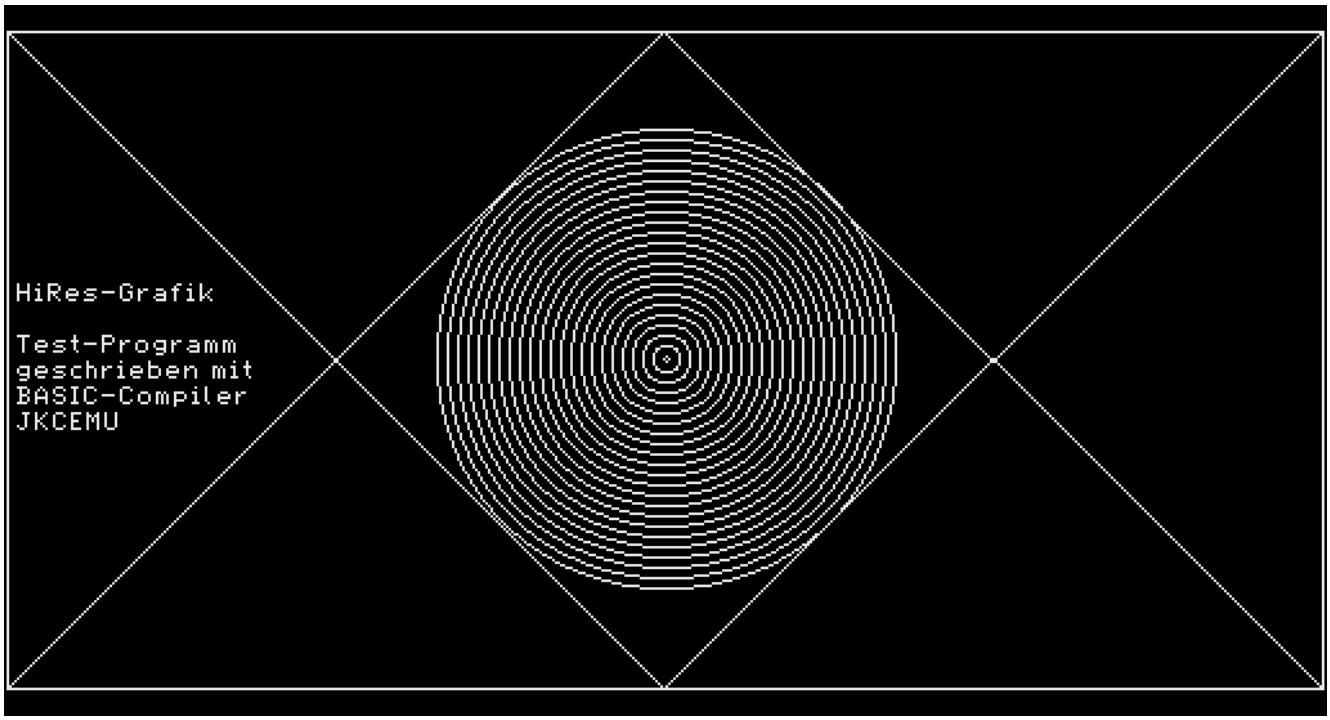
Die LLC2-HiRes-Erweiterung

Der „normale“ Bildschirmmodus des LLC2 hat 32 Zeilen zu je 64 Zeichen. Die einzelnen Zeichen bestehen aus 8x8 Pixel, sind also im Gegensatz zum AC1 quadratisch und ergeben damit ein etwas breiteres Bild (2:1). Eine Pseudografik von 128x64 „Pixel“ ist möglich und wird auch von GS-BASIC unterstützt. Der Zeichengenerator enthält dazu:

$000…$3FF Grundzeichensatz
$400…$7FF Pseudografikzeichen (Codes $80...$FF)

Mittels dreier Schaltkreise wurde eine Erweiterung für eine schwarz-weiß Vollgrafik mit einer Auflösung von 512x256 Pixeln entwickelt (vgl. Anlage 1). Diese so genannte "HiRes"-Erweiterung nutzt statt des originalen 2k-Bildwiederholspeichers ($C000…$C7FF) satte 16 KByte im Arbeitsspeicher des LLC2. Der Zeichengenerator ist um eine 2kByte große Bitmap zu ergänzen. Damit kann nun jedes Pixel auf dem Bildschirm einzeln adressiert werden.

Mit dieser Technik lassen sich viel feinere Zeichnungen erstellen, als es mit der Pseudo-Grafik möglich ist:



Testgrafik (Das zugehörige JKCEMU-BASIC-Listing ist in Anlage 2 dargestellt.)

Der „neue“ LLC2 hat den HiRes-Zusatz bereits auf der Hauptplatine integriert. Im praktischen Umgang merkt man jedoch nichts davon. Die meisten Programme (z.B. Monitor, GS-Basic[1], …) arbeiten alle nur mit dem Standard-BWS.

Den HiRes-Modus können nur die dafür vorgesehene Programme nutzen. Bekannte Anwendungen sind z.B.:

- „Grafikplotter“
- „ELOCAD“ (Schaltplanzeichenprogramm)
- „Vollgrafik“ (Bibliothek von Grafikroutinen mit Test- und Demoprogramm)
- ein Treiber unter CP/L, mit dem statt der 64Zeichen/32Zeilen das 80Zeichen/24Zeilen-Format per Grafik dargestellt werden kann.

[1] Allerdings ist aus GS-BASIC heraus der Umgang mit HiRes möglich, siehe unten!

Die LLC2-HiRes-Erweiterung

Mit der HiRes-Erweiterung muss man nun zwischen dem herkömmlichen ***Textmodus (ASCII-Mode)*** und dem ***Grafikmodus*** unterscheiden. Die Umschaltung zwischen beiden Modi erfolgt über das Steuerregister $EE. Im Zusammenhang mit dem Modul2 (Farbgrafik) hat dieses Register noch andere Funktionen, die hier jedoch nicht weiter betrachtet werden sollen. Durch ein entsprechendes Setzen verschiedener Bits in diesem Register lassen sich die Eigenschaften der Bilddarstellung variieren.:

Die Bits des Steuerregister $EE wirken wie folgt (ohne Modul 2 Steuereigenschaften):

Bit	7	6	5	4	3	2	1	0
Textmodus	0	**0**	0		0	**Textbank 1 oder 2**	0	0
HiRes-Modus	0	**1**	**Grafik-Bank 1-4**		0	0		

Bit 6 entscheidet also, welcher Modus verwendet wird. Bit2 im Textmodus und Bit4+5 im HiRes-Modus bestimmen den benutzten Speicherbereich. Es ergibt sich damit folgende Zuordnung:

Speicherbereich	**Bemerkung**	**Steuer-Byte**
Textspeicher 1 : $C000 - $C7FF	Standard, Text und Pseudografik	00h
Textspeicher 2 : $F800 - $FFFF	nur ASCII-Mode	04h
Grafikspeicher 1: $C000 - $FFFF		40h
Grafikspeicher 2: $8000 - $BFFF	von BASIC-Compiler des JKCEMU verwendet!	50h
Grafikspeicher 3: $4000 - $7FFF		60h
Grafikspeicher 4: $0000 - $3FFF		70h

Textspeicher 1 ist der normale BWS. Hier landen alle Schirm-Ausgaben des Monitors, die von GS-Basic oder anderer auf den Schirm „druckenden“ Programme.

Textspeicher 2 entspricht diesem und ist ein reiner Text- (und Pseudografik-)Bildschirm. Das Beschreiben muss jedoch „zu Fuß“ erfolgen (z.B. POKE). Die ab $F800…$FFFF abgelegten Zeichen erscheinen auch in dieser Reihenfolge auf dem Schirm. Die Nutzung des Textspeicher 2 bietet sich an, um beispielsweise im Hintergrund etwas aufzubauen und dann schlagartig dorthin zu schalten.

Die Grafik-Speicher 1 bis 4 lassen die Darstellung von HiRes-Grafiken zu, jedoch kein normales Beschreiben mit ASCII-Zeichen. Diese müssen „emuliert“ werden, d.h. jedes Zeichen ist in eine Gruppe von 8*8 Pixel umzurechnen und dann als Grafik darzustellen.

Eine kombinierte Betriebsart (HiRes-Grafik und ASCII-Zeichen auf je einem halben Bildschirm) ist im („Split-Screen-Modus“) möglich. Diesem Zweck dient neben der 3-IS-Hardware-Erweiterung die Brücke von D4/6 zu D33/21. Zur Nutzung dieser Betriebsart muss der CTC-Baustein entsprechend programmiert werden, sodass nach der Hälfte des Bildschirmes „automatisch“ der Bildschirm von Text- auf Grafikmodus bzw. umgekehrt umgeschaltet wird.

Die LLC2-HiRes-Erweiterung

Organisation des Grafik-BWS:

Für je 8 horizontal aufeinander folgende Pixel auf dem Grafikschirm ist ein Byte im RAM zuständig. Ein gesetztes Bit in diesem Byte bedeutet „Pixel sichtbar“. Dabei ist die umgekehrte Zuordnung zu beachten:

Byte=01h => Pixel 8 gesetzt … 80h => Pixel 1 gesetzt

Der Zeichengenerator ist um 2K zu erweitern und enthält im Bereich von $800-$FFF die möglichen Bitmuster für jeweils 8 Pixel. Aus Gründen der einfachen Hardware der HiRes-Erweiterung werden immer 8 aufeinander folgende gleiche Muster benötigt (das entspricht den 8 Pixelzeilen eines Zeichens im Normalmodus), womit der ZG quasi überbrückt wird.

Ebenfalls der Hardware ist geschuldet, dass fortlaufende Adressen nicht fortlaufende Pixel bedeuten. Am Beispiel Grafikspeicher 2 ($8000…$BFFF), auch vom BASIC-Compiler des JKCEMU verwendet, ergibt sich folgende Zuordnung:

Der gelb markierte Bereich entspricht einem Standard-Zeichen im ASCII-Mode (8x8Pixel).

Spalte>	1	2		64
Pixelzeile v	Pixel 1-8	Pixel 9-16		Pixel 504-512
1	$8000	$8001		$803F
2	$8800	…		…
3	$9000			
4	$9800			
5	$A000			…
6	$A800			
7	$B000			
8	$B800			
9	$8040			
256	$BFC0	$BFC1		$BFFF

Beachte:
Der Ursprung x=y=0 bei Verwendung von Grafikfunktionen (z.B. PSET x,y) mit dem JKCEMU-BASIC-Compiler liegt links unten (Anfangsadresse $BFC0)!

Die LLC2-HiRes-Erweiterung

Nutzung des HiRes-Modus

Anwendungen für die Nutzung der HiRes-Grafik erfordern eine spezielle Programmierung bzw. Technik. Grundsätzlich erfolgt die Steuerung so:

1. je nach Anwendung den Grafikspeicher planen/festlegen (z.B. $8000 - $BFFF)
2. Grafik-Modus entsprechend einschalten:

```
LD A,50          ; Grafikspeicher 2 ($8000 - $BFFF) einstellen
OUT (EEH),A      ; Register HiRes
```

3. in den RAM-Bereich das Gewünschte zeichnen…
4. zu Programm-Ende den Grafikmodus wieder ausschalten:

```
LD A,0           ; Standard-BWS
OUT (EEH),A      ; Register HiRes
JP  07FDH        ; GETCO => zurück Monitor
```

Zeichnet man vor der Umschaltung in den Bild-RAM und schaltet erst dann den Modus um, so erfolgt die Darstellung schlagartig. Ansonsten kann man (je nach Umfang der Zeichenfunktionen) den allmählichen Bildaufbau beobachten.

Das Ansprechen des Grafikspeicher ist also nicht ganz so unkompliziert wie das Schreiben von ASCII-Zeichen auf dem Standard-BWS. Um ein bestimmtes Pixel zu verändern muss man prinzipiell:

- die zugehörige BWS-Adresse wissen/bestimmen,
- ein komplettes Byte an dieser Stelle auslesen,
- den Wert entsprechend gewünschtem Pixel modifizieren und
- das Byte zurückschreiben.

Wie zeichnet man unter HiRes?

Das Setzen (bzw. Löschen oder Invertieren) eines Pixels ist die elementarste Zeichenfunktion. Für das Zeichnen von Linien und Kreisen sind weitere Algorithmen (z.B. nach BRESENHAM) nötig. Es ist leicht einzusehen, dass dies einigen Rechenaufwand erfordert, sodass schon aus Zeitgründen die Nutzung von Maschinencode erforderlich ist.

Auch für das Einbringen von Text in die Grafik bedarf es spezieller Routinen, da ein Textzeichen erst in ein 8x8Pixel-Grafikzeichen umgerechnet werden muss.

Es gibt dafür fertige Bibliotheken mit den wichtigsten Funktionen. ***Torsten Schönitz aus Dresden*** hatte 1988 eine umfangreiche Bibliothek mit verschiedenen Routinen geschrieben, die er als „***Vollgrafik-Treiber***“ zur kostenlosen Nutzung bereitstellte. Damit lassen sich unter Angabe der nötigen Koordinaten und Aufruf der entsprechenden Teilroutine die gewünschten Figuren zeichnen sowie im Textmodus „beschriften“

Aber auch aus BASIC heraus ist die Nutzung von HiRes möglich. Wie das geht, zeigt das nachfolgende Beispiel mit GS-BASIC. Nichts anderes macht übrigens das Elektronik-Zeichenprogramm ELOCAD, nur dass eine andere Grafik-Bibliothek verwendet wird. Auch hier muss diese als erstes geladen werden. In Basic werden die Parameter berechnet und die MC-Funktionen aufgerufen...

Beispiel in GS-BASIC

Der Zugriff aus BASIC heraus ist leicht möglich, vorausgesetzt, dass die Maschinencode-Routinen (="Treiber") geladen wurden. Der Vollgrafik-Treiber von Torsten Schönitz ist vor dem BASIC-Start zu laden; er belegt den Speicherplatz von $2000 bis $3948 [2]. Für das folgende Beispiel werden nur ein paar Grundroutinen daraus genutzt:

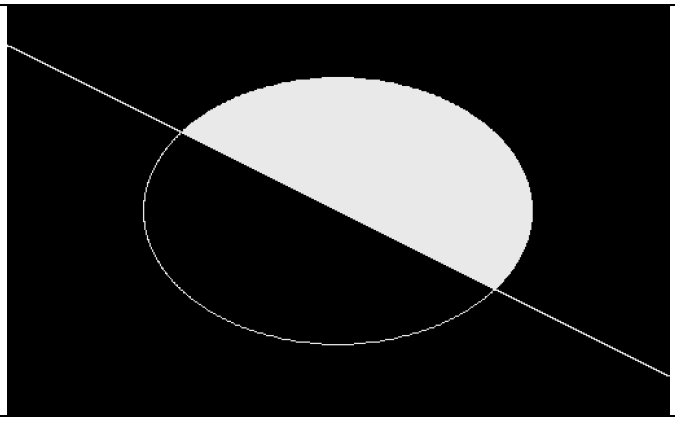
Funktion	Aufruf	Parameterübergabe
Linie Zwischen zwei Punkten zeichnen	$2F8A	X-Koordinate Punkt 1 in $2098 (=8344) Y-Koordinate Punkt 1 in $209A (=8346) X-Koordinate Punkt 2 in $2088 (=8328) Y-Koordinate Punkt 2 in $208A (=8330)
Kreis zeichnen	$2DD2	X-Koordinate Mittelpunkt in $2060 (=8288) Y-Koordinate Mittelpunkt in $2062 (=8290) Radius in $2064 (=8292)
Füllen	$2FE8	füllt die Fläche um den aktuellen Grafikkursor
Grafik-„CLS"	$32D2	Grafik-Bildschirm löschen
Grafikkursor positionieren	-	X-Koordinate in $2088 (=8328) Y-Koordinate in $208a (=8330)

Koordinaten: 0<X<511 und 0<Y<255, Ursprung X=Y=0 ist links oben!

```
 10 CLEAR 500,32767                    :REM RAM-Ende begrenzen (<$8000)
 20 CALL*32D2                          :REM Grafikbereich löschen
 30 OUT 238,80                         :REM umschalten auf Grafikmode  ($8000)
 40 X1=0:Y1=0:X2=511:Y2=255:GOSUB100:REM Linie zeichnen
 50 X1=255:Y1=128:R=150:GOSUB120       :REM Kreis zeichnen
 60 X1=355:Y1=80:GOSUB90               :REM Grafikkursor setzen
 70 CALL*2FE8                          :REM rundherum füllen
 80 GOTO 140
 90 DOKE 8328,X1:DOKE 8330,Y1:RETURN:REM neue Grafik-Koordinaten
100 DOKE 8344,X1:DOKE 8346,Y1:DOKE 8328,X2:DOKE 8330,Y2   :REM Koordinaten
110 CALL*2F8A:RETURN                   :REM Linie
120 DOKE 8288,X1:DOKE 8290,Y1:DOKE 8292,R                 :REM Koordinaten
130 CALL*2DD2:RETURN                   :REM Kreis
140 INKEY A$:IF A$=""THEN 170          :REM warten auf eine Taste
150 OUT 238,0                          :REM zurück zum Textmode (BASIC)
```

Damit sollte sich folgender Bildschirminhalt ergeben:

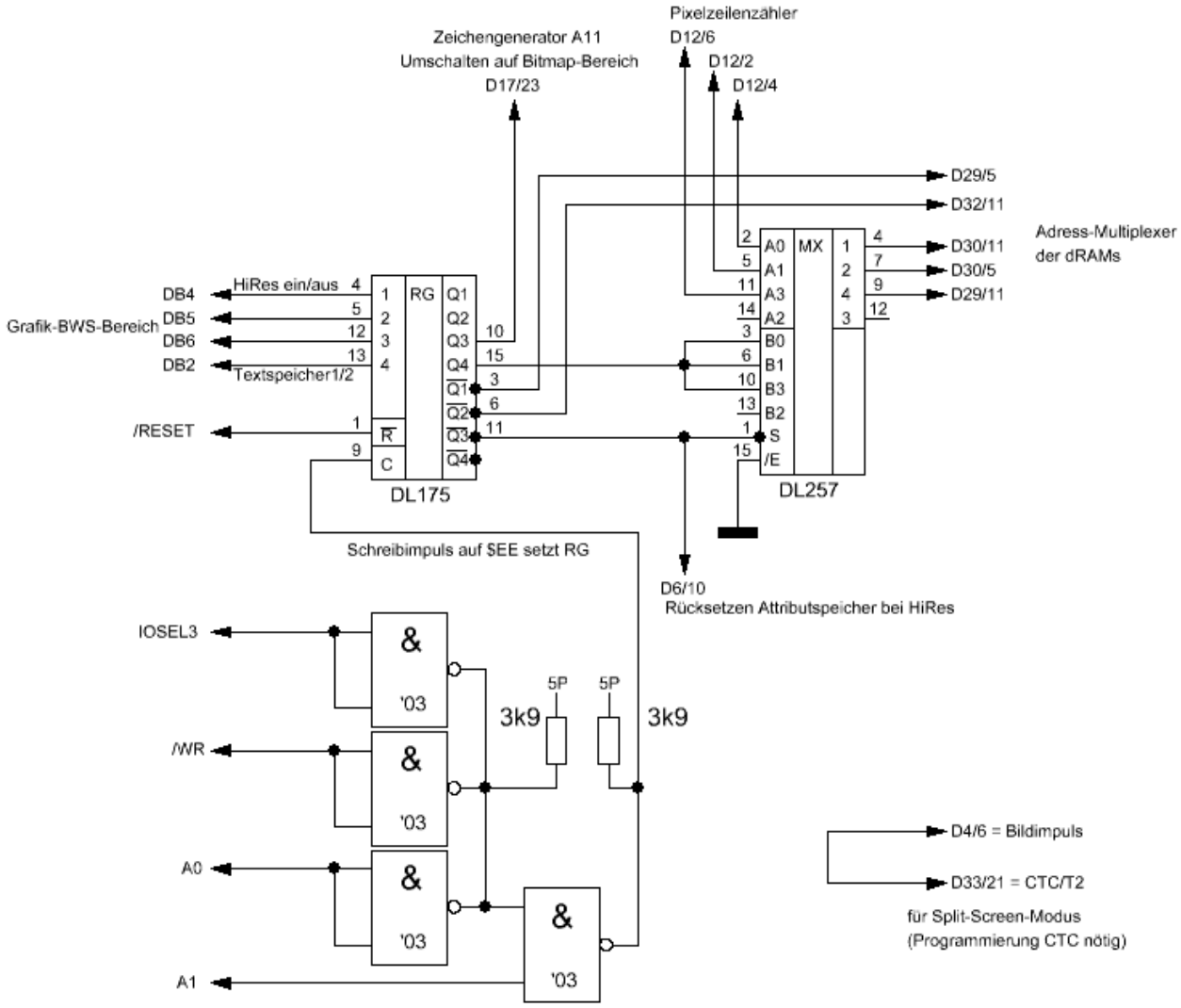
Je nach Bildschirmeinstellung ist der Kreis mehr oder weniger eine Ellipse, so wie hier bei dem mit JKCEMU erzeugten Bild...



Wichtig ist die Begrenzung des für BASIC verfügbaren RAM auf <$8000, denn ab dort beginnt (in diesem Beispiel) der Grafikspeicher!

[2] Im Original etwas länger, auf die einzelnen Unterprogramme folgt noch ein Test-/Demoprogramm.

Anlage 1:
Schaltplan



Bitte beachten:
Die auf der Hauptplatine befindliche Brücke Br2 darf nicht gesteckt werden, da die HiRes-Grafikerweiterung auf der Platine bereits angeschlossen ist und damit A11 schon mit einem Ausgang gesteuert wird! Mit Br2 erfolgte in der Originalversion die Zeichengenerator-Umschaltung (A11 des ZG-Eproms entweder auf 5P oder PIO B2).

Anlage 2:
Quellcode für die Testgrafik im JKCEMU-Format

```
SCREEN LASTSCREEN
CLS

FOR I=1 TO 90 STEP 4
CIRCLE 256,128,I
NEXT

LINE 0,0,0,255
LINE 0,255,511,255
LINE 511,255,511,0
LINE 511,0,0,0

LINE 0,0,255,255
LINE 255,255,511,0

LINE 0,255,255,0
LINE 255,0,511,255

MOVE 0,150
LABEL " HiRes-Grafik"
MOVE 0,130
LABEL " Test-Programm"
MOVE 0,120
LABEL " geschrieben mit"
MOVE 0,110
LABEL " BASIC-COMPILER"
MOVE 0,100
LABEL " JKCEMU"

PAUSE 40
CLS
SCREEN 0
```

Der BASIC-Compiler des JKCEMU erzeugt daraus ein knapp 2 KByte großes Binärfile, welches ab Adresse $2000 geladen und direkt am LLC2 gestartet werden kann. Das Programm stellt die eingangs gezeigte Grafik dar und kehrt nach einer Pause von ca. 4 sek. wieder zum Monitor zurück.

Erstellt von:
RolfWeidlich@web.de
Stand: 18.04.2013