

## Portbelegungen der GDP64HS-FPGA

### SPI:

\$FFFFFFF00 - SPI-Control-Register

#### Schreiben

7	6	5	4	3	2	1	0	
								----- Clockdivider
								000 = 40MHz/2 => 20,0MHz
								111 = 40MHz/16 => 2,5MHz
								----- frei
								----- frei
								----- Slave Select
								01 = Slave 0, 10 = Slave 1
								----- SPI-Controller enable

#### Lesen

7	6	5	4	3	2	1	0	
								--- IDLE 1 = Controller bereit für Daten
								----- Write Collision 1 = Datenverlust
								----- frei

\$FFFFFFF01 - SPI-Daten-Register

---

### SOUND:

\$FFFFFFF50 - \$FFFFFFF51

Funktion wie SOUND-Baugruppe

---

### GDP:

\$FFFFFFF60 - Seitenumschaltung / XOR / DMA

#### Schreiben

7	6	5	4	3	2	1	0	
								--- XOR-MODE (1 = aktiv)
								----- frei
								----- Leseseite (0 - 3)
								----- Schreibseite (0 - 3)

#### Lesen

Ergebnis der DMA-Abfrage (nach Befehl \$0f an GDP)  
Das Ergebnis ist invertiert!  
Die Pixeladresse wird in den X- und Y-Registern übergeben  
Bei der S/W-Variante werden 8 Pixel geliefert,  
bei der Farb-Variante 2

\$FFFFFFF61 - Scrollport

#### Schreiben

Hardware-Scrollwert in 2er Schritten (LSB wird ignoriert)

## GDP-Register

\$FFFFFFF70 - Status / CMD

\$FFFFFFF71 - CTRL 1

\$FFFFFFF72 - CTRL 2

7	6	5	4	3	2	1	0	
								----- Art der Vektoren
								----- 1 = Buchstaben kursiv
								----- 1 = Buchstaben vertikal
								----- 1 = USER-Zeichensatz
								----- 1 = Transparent-Mode
								----- frei

## USER-Zeichensatz

Der USER-Zeichensatz befindet sich im RAM des FPGA, dieses ist mit dem deutschen NKC-Zeichensatz vorbelegt. Es stehen 96 Zeichen zur Verfügung. Diese beginnen bei dem Zeichen \$20 entsprechend SPACE und enden mit dem Zeichen \$7F.

### Schreiben eines neuen Zeichens:

1. Setzen des Bits 4 in CTRL2
2. Laden der 1. Zeichenadresse in die X-Register (\$FFFFFFF78 + 79)  
Diese Adresse errechnet sich wie folgt:  
(NR-\$20)\*5  
NR = ASCII-Nummer des Zeichens (z.B. \$41 fürs 'A')
3. Schreiben von 5 Byte mit dem Muster des Zeichen auf Port \$FFFFFFF7E  
Das Muster des Zeichens ist vom Aufbau wie bei dem Befehl PROGZGE.  
Dies erfolgt ohne erneutes Schreiben des X-Registers (autoincrement).

z.B. das Omega: dc.b %10011110  
dc.b %11100001  
dc.b %00000001  
dc.b %11100001  
dc.b %10011110

## Transparent-Mode

Wie bei der original GDP werden beim Schreiben eines Buchstaben nur die Pixel mit der Vordergrundfarbe neu geschrieben. Daher bekommt man "Buchstabensalat", wenn zwei oder mehr Zeichen auf der selben Position geschrieben werden.

Anders wenn man das Transparent-Mode-Bit setzt, dann werden auch die Pixel mit Hintergrundfarbe neu geschrieben. Man kann dann also auf ein Löschen verzichten, bevor ein neues Zeichen geschrieben wird.

\$FFFFFFF73 - CSIZE

\$FFFFFFF74 - DeltaX MSB (1 Bit) neu ab Version vom 02.05.09

\$FFFFFFF75 - DeltaX LSBs (8 Bit)

\$FFFFFFF76 - DeltaY MSB (1 Bit) neu ab Version vom 02.05.09

\$FFFFFFF77 - DeltaY LSBs (8 Bit)

\$FFFFFFF78 - X MSBs (4 Bit)

\$FFFFFFF79 - X LSBs (8 Bit)

\$FFFFFFF7A - Y MSBs (4 Bit)

\$FFFFFFF7B - Y LSBs (8 Bit)

\$FFFFFF7C - XLP / Reserve  
\$FFFFFF7D - YLP / Reserve

\$FFFFFF7E - Schreibport für USER-Zeichensatz

siehe CTRL 2

\$FFFFFF7F - Reserve

#### GDP Farberweiterung

\$FFFFFFA0 - Vordergrundfarbe

Farbnummer der Vordergrundfarbe  
siehe CLUT

\$FFFFFFA1 - Hintergrundfarbe

Farbnummer der Hintergrundfarbe  
siehe CLUT

\$FFFFFFA4 - CLUT Farbnummer

\$FFFFFFA5 - CLUT Daten MSB (1 Bit)

\$FFFFFFA6 - CLUT Daten LSBs (8 Bit)

Die CLUT (ColorLookUpTable) ist eine Farbtabelle mit 16 9-Bit-Einträgen. Dies bietet die Möglichkeit gleichzeitig 16 der 512 möglichen Farben darzustellen. Die darzustellende Farbe wird durch ihre Nummer in der CLUT festgelegt (\$FFFFFFA0 + A1). In der unten stehenden Tabelle ist die Standardbelegung aufgeführt.

Zum ändern einer Farbe muss zunächst die gewünschte Farbnummer in das Register \$FFFFFFA4 geschrieben werden, danach das Highbyte und dann das Lowbyte des Farbwertes in die Register \$FFFFFFA5 und A6.  
Es können weitere Farbwerte, ohne erneutes schreiben des Registers \$FFFFFFA4, übertragen werden (autoincrement).

#### Standardbelegung der CLUT

Farbnummer dez	hex	Name NKC	I-Net	Wert hex	binär			Entsprechender HTML-Code
					R	G	B	
0	\$0	Schwarz	black	\$0000	%000	000	000	#000000
1	\$1	Weiß	white	\$01FF	%111	111	111	#FFFFFF
2	\$2	Gelb	yellow	\$01F8	%111	111	000	#FFFF00
3	\$3	Grün	lime	\$0038	%000	111	000	#00FF00
4	\$4	Rot	red	\$01C0	%111	000	000	#FF0000
5	\$5	Blau	blue	\$0007	%000	000	111	#0000FF
6	\$6	Violett	fuchsia	\$01C7	%111	000	111	#FF00FF
7	\$7	Zyan	aqua	\$003F	%000	111	111	#00FFFF
8	\$8	Dunkelgrau	gray	\$0092	%010	010	010	#404040
9	\$9	Hellgrau	silver	\$0124	%100	100	100	#808080
10	\$A	Dunkelgelb	olive	\$00D8	%011	011	000	#606000
11	\$B	Dunkelgrün	green	\$0018	%000	011	000	#006000
12	\$C	Dunkelrot	maroon	\$00C0	%011	000	000	#600000
13	\$D	Dunkelblau	navy	\$0003	%000	000	011	#000060
14	\$E	Violett dunkel	purple	\$00C3	%011	000	011	#600060
15	\$F	Zyan dunkel	teal	\$001B	%000	011	011	#006060

-----

KEY:

\$FFFFFF68 - \$FFFFFF69

Funktion wie Key-Karte

-----

MAUS:

\$FFFFFF88 - \$FFFFFF8F

Funktion wie HARDCOPY/MAUS-Baugruppe

-----

TIMER:

\$FFFFFFF4 - Control Register

7	6	5	4	3	2	1	0	
								---
								Run (Timer run)
								----- WRM (Timer Register Write Mode)
								0=Write only reload register
								1=Write only timer register
								2=Write both
								----- frei
								----- TOVF Timer Overflow Flag.
								Muss per Software zurückgesetzt werden
								----- IE (Interrupt enable)

\$FFFFFFF5 - Counter/Reload Register High

\$FFFFFFF6 - Counter/Reload Register Low

Das Counter und Reload Register ist je 16 bit breit.

Der Timer zählt mit 1 MHz abwärts.

Wenn das High-byte beschrieben wird dann wird der Wert in einem Zwischenregister gespeichert.

Wird dann das Low-byte geschrieben wird dieser Wert zusammen mit dem zwischengespeicherten High-Byte in das Counter oder Reload (oder beide) übernommen (somit werden immer alle 16 bit auf einmal geschrieben).

Gelesen wird immer das Counter-Register.

Für den Interrupt-Betrieb muss der Jumper auf der FPGA-Karte eingesetzt werden.