

Dokumentation UCT

Dok-Rev. 1.2 vom 24.08.2009
Hardware-Rev. 1.0 vom 09.03.2009



Inhaltsverzeichnis

Sicherheit	4
2 Allgemeine Hinweise.....	6
2.1 Handhabung	6
2.2 Installation	6
2.3 Erklärung	6
2.4 Reparaturen	6
3 Technische Daten.....	7
3.1 Umgebungsbedingungen	7
3.2 Mechanische Abmessungen	7
3.3 Technische Daten	7
4 Inbetriebnahme.....	8
4.1 Einbau	8
4.2 Anschlußkabel	8
4.3 Spannungsversorgung	9
4.4 Lage der Jumper und Anschlüsse	10
4.5 Beschreibung der Jumper	10
4.5.1 ST14/15: Analoge Ausgänge Strom/Spannung umschalten	10
5 Hardwarebeschreibung.....	11
5.1 Serielle Schnittstellen A1	11
5.1.1 Belegung A1	11
5.2 CAN-Bus	11
5.2.1 Belegung CAN-Bus	11
5.3 LCD	11
5.3.1 Belegung LCD-Anschluß	12
5.4 Leistungsausgänge	12
5.4.1 Solid State Relais	12
5.4.2 Relais 230V	12
5.4.3 Relais potenzialfrei	12
5.5 Digitale Eingänge	13
5.6 Analoge Ausgänge	13
5.7 Analoge Eingänge	13
5.8 Leuchtdioden	14
5.9 Batterie/Goldcap	14
5.10 Socket Modem	14

6	Programmierung	15
6.1	Adreßbelegung	15
6.2	Interruptquellen	15
6.2.1	Events	15
6.3	Watchdog	16
6.4	Ein-/Ausgänge	16
6.4.1	Digitale Eingänge	16
6.4.2	Analoge Ein-/Ausgänge, SSR, Relais	16
6.4.2.1	<i>Beschreibung der Strukturelemente</i>	17
6.5	Serielle Schnittstelle	18
6.6	Socket Modem	18
7	Update RTOS-UH	19
7.1	FLASH-Belegung	19
7.2	RTOS-UH ohne Anwender-Flash starten	19
8	Terminalemulation	20
8.1	Allgemeines	20
8.2	Befehlsvorrat Terminalemulation	20
8.3	Umsetzung der Eingabezeichen	22
8.4	Grafikfunktionen	23
8.4.1	Funktionsumfang und Implementierungsabhängigkeiten	24
8.4.2	Allgemeines	25
8.4.3	Funktionsreferenz	26
8.4.4	Zeichensatz	31

Revisionsliste:

Rev.	Datum	Na.	Änderung
1.0	29.06.2009	Ko	Erstellung
1.1	31.07.2009	Ko	Fonts ergänzt
1.2	24.08.2009	Ko	Socket-Modem ergänzt

1 Sicherheit

Gefahr!



Lebensgefährliche Betriebsspannung!

Lebensgefahr durch Stromschlag!

Vor Arbeiten am UCT ist die Spannungsversorgung abzuschalten und gegen Wiedereinschalten zu sichern.

Gefahr!



Nässe und Flüssigkeiten aus der Umgebung können ins Innere des Gerätes gelangen.

Lebensgefahr durch Stromschlag bei Berührung!

Das UCT darf nicht in nassen oder feuchten Umgebungen oder direkt in der Nähe von Gewässern eingesetzt werden. Installieren Sie das Gerät an einem trockenen, vor Spritzwasser geschützten Ort

Gefahr!



Überspannung, Überstrom.

Brandgefahr!

Sichern Sie das UCT gegen Überspannung ab. Verwenden Sie nur passende Sicherungen.

Warnung!



Kurzschlüsse und Beschädigung durch unsachgemäße Reparaturen und Öffnen von Wartungsbereichen.

Feuer, Funktionsausfall und Verletzungsgefahr!

Nur ausgebildetes Personal darf das UCT öffnen und Arbeiten ausführen.

Hinweis

Beschädigung des Touch Bildschirms!

Harte Gegenstände können Kratzer auf dem Display hinterlassen oder es zerstören.

Benutzen Sie zur Bedienung nur den Finger oder PDA-Stifte aus Kunststoff.

Hinweis

Beschädigung durch Chemikalien!

Ketone oder chlorierte Kohlenwasserstoffe lösen den Kunststoff des Gehäuses an und beschädigen die Oberfläche.

Bringen Sie das UCT auf keinen Fall mit z.B. Aceton oder z.B. Dichlormethen in Berührung.

2 Allgemeine Hinweise

2.1 Handhabung

1. Lesen Sie bitte zuerst sorgfältig diese Dokumentation bevor Sie die Hardware auspacken und einschalten. Sie sparen Zeit und vermeiden Probleme.
2. Beachten Sie bitte die Vorsichtsmaßnahmen bei der Handhabung elektrostatisch gefährdeter Hardware.
3. Wenn die Hardware Batterien enthält, legen Sie sie nicht auf elektrisch leitfähige Unterlagen. Die Batterie könnte kurzgeschlossen werden und Schäden verursachen.
4. Achten Sie bitte darauf, daß der spezifizierete Temperaturbereich nicht verlassen wird.

2.2 Installation

1. Überprüfen Sie, ob alle Jumper entsprechend Ihrer Anwendung gesetzt sind.
2. Schalten Sie die Spannungsversorgung der externen Anschlüsse ab, bevor Sie eine Verbindung herstellen.
3. Wenn Sie sicher sind, daß alle Verbindungen korrekt installiert sind, schalten Sie die Spannungsversorgung ein.

2.3 Erklärung

Wir behalten uns das Recht vor, Änderungen, die einer Verbesserung der Schaltung oder des Produktes dienen, ohne besondere Hinweise vorzunehmen. Trotz sorgfältiger Kontrolle kann für die Richtigkeit der hier gegebenen Daten, Schaltpläne, Programme und Beschreibungen keine Haftung übernommen werden. Die Eignung des Produktes für einen bestimmten Einsatzzweck wird nicht zugesichert.

2.4 Reparaturen

Sollte das Produkt defekt sein, so senden Sie es bitte frei in geeigneter Verpackung mit folgender Beschreibung an uns zurück:

- Fehlerbeschreibung
- Trat der Fehler nur unter bestimmten Bedingungen auf?
- Was war angeschlossen?
- Wie sahen die angeschlossenen Signale aus?
- Garantiereparatur oder nicht?

3 Technische Daten

3.1 Umgebungsbedingungen

Umgebungstemperatur (Betrieb)	0-50° C
Umgebungstemperatur (Lagerung)	-20-85° C
rel. Luftfeuchte	max. 95%, nicht kondensierend
Höhe	-300m bis +3000m

3.2 Mechanische Abmessungen

Gehäusegröße	260 x 230 x 120 mm
Schutzart	IP 40
Anschlüsse	Zugfederklemmen

3.3 Technische Daten

Versorgungsspannung:	230 Volt ~, max. 16 A
Prozessor	MC68332
Batteriepufferung:	für Echtzeituhr und RAM, über Lithium-Batterie oder Gold-Cap-Kondensator
Analogeingänge	16 differenzielle, unipolar, Eingangsbereich anpassbar, Auflösung 10 Bit, default: 14 x Pt1000 (-30..120°C), 0..10V, 0..20mA
Analogausgänge	2 Stück, 0..20mA oder 0-10 Volt über PWM
Digitaleingänge:	8 Stück, 24 Volt über Optokoppler
Leistungsausgänge:	4 Stück SSR 230V / 2 A 4 Stück Relais 230V / 2 A 2 Stück Relais Schließer potentialfrei 2 Stück Relais Umschalter potentialfrei
Bedienung:	AMOLED-Display 2,8", 320 x 240 Pixel, 15 Bit Farbauflösung Touchscreen
Ser. Schnittstellen	1x 5-Draht RS232
CAN	1x CAN galvanisch getrennt
Option	Stecksocket für Modem, GSM, Ethernet

4 Inbetriebnahme

4.1 Einbau

Das **UniverselleCanTerminal** muß EMV-gerecht von entsprechend ausgebildetem Fachpersonal verkabelt werden. Das Gerät wird mit Netzspannung versorgt!

4.2 Anschlußkabel

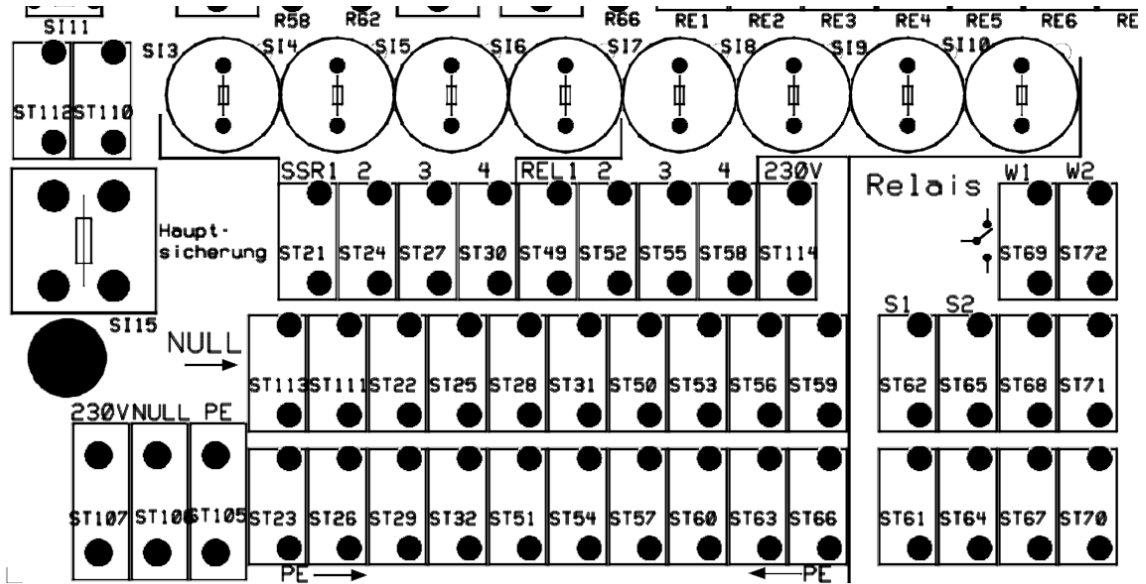
Die Anschlußkabel (außer der Einspeisung) müssen folgende Bedingungen einhalten:

Leiterquerschnitt starr min	0,2 mm ²
Leiterquerschnitt starr max	1,5 mm ²
Leiterquerschnitt flexibel min	0,2 mm ²
Leiterquerschnitt flexibel max	1,5 mm ²
Leiterquerschnitt flexibel m. Aderendhülse ohne Kunststoffhülse min	0,25 mm ²
Leiterquerschnitt flexibel m. Aderendhülse ohne Kunststoffhülse max	0,75 mm ²
Leiterquerschnitt flexibel m. Aderendhülse m. Kunststoffhülse min	0,25 mm ²
Leiterquerschnitt flexibel m. Aderendhülse m. Kunststoffhülse max	0,75 mm ²

Für die Einspeisung sind folgende Querschnitte zulässig:

Leiterquerschnitt starr min	0,2 mm ²
Leiterquerschnitt starr max	4 mm ²
Leiterquerschnitt flexibel min	0,2 mm ²
Leiterquerschnitt flexibel max	2,5 mm ²
Leiterquerschnitt flexibel m. Aderendhülse ohne Kunststoffh. min	0,25 mm ² Abisolierlänge 8 mm
Leiterquerschnitt flexibel m. Aderendhülse ohne Kunststoffh. max	2,5 mm ² Abisolierlänge 8 mm
Leiterquerschnitt flexibel m. Aderendhülse m. Kunststoffhülse min	0,25 mm ² Abisolierlänge 8 mm
Leiterquerschnitt flexibel m. Aderendhülse m. Kunststoffh. max	1,5 mm ² Abisolierlänge 8 mm

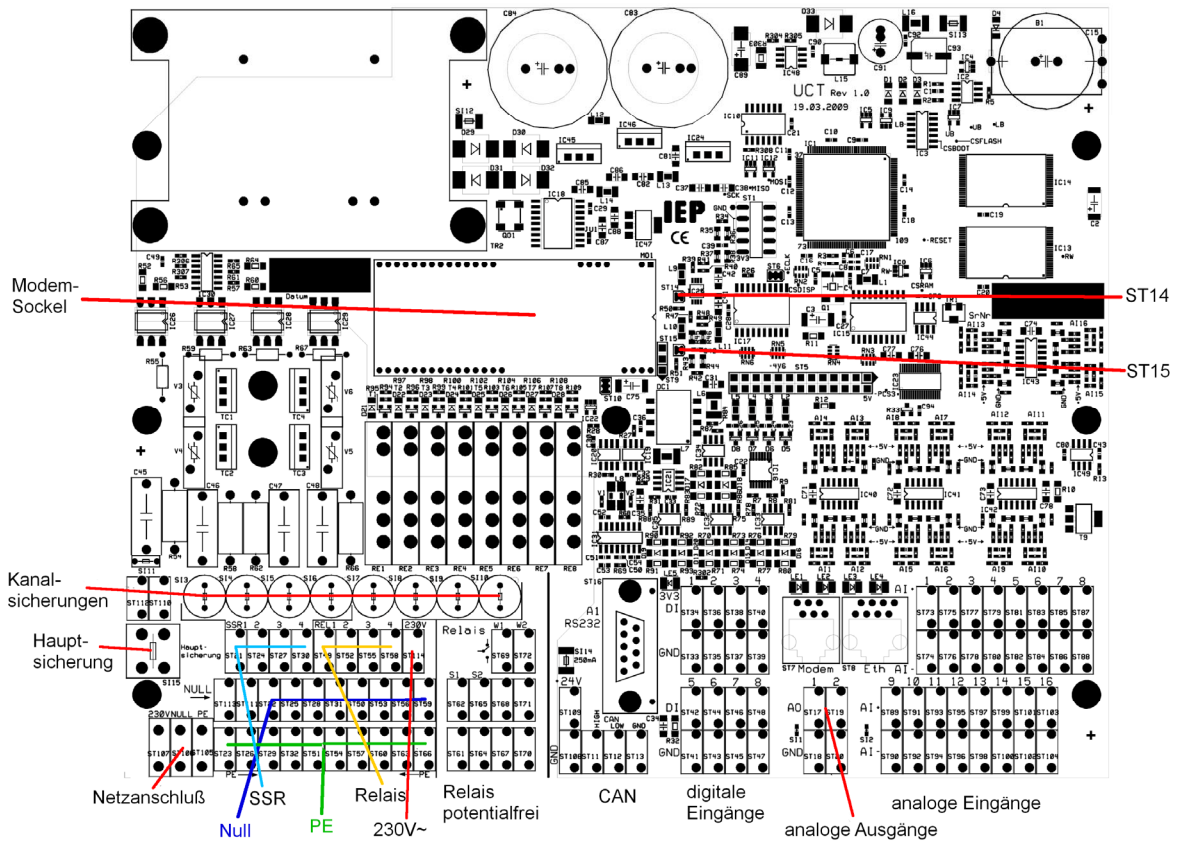
4.3 Spannungsversorgung



Die Spannungsversorgung ist für 230 Volt ~ ausgelegt. Die Phase ist an Klemme ST107 (230V), der Nullleiter an ST106 (NULL) und der Schutzleiter an ST105 (PE) anzuschließen.

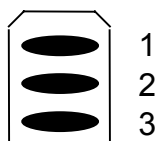
Die untere Reihe Klemmen (ST23 – ST66) ist durchverbunden und mit PE belegt. Die mittlere Reihe Klemmen (ST22 – ST59) ist ebenfalls durchverbunden und mit dem Nullleiter über den Ein-/Ausschalter verbunden. Der Ein-/Ausschalter wird an den beiden linken Klemmen (ST113/ST111) angeschlossen. Der 2.te Pol des Ein-/Ausschalters schaltet die Phase und wird an ST112/110 angeschlossen.

4.4 Lage der Jumper und Anschlüsse



4.5 Beschreibung der Jumper

Die Löt-Jumper werden folgendermaßen gezählt:



4.5.1 ST14/15: Analoge Ausgänge Strom/Spannung umschalten

Die analogen Ausgänge können 0-20mA liefern. Wird der entsprechende Jumper geschlossen, so wird eine Bürde von 510 Ω auf den Ausgang geschaltet, so dass 0-10V ausgegeben werden.

Ausgang	Jumper
0-20 mA	offen
0-10 V	1-2

Die Zuordnung der Jumper zu den Ausgängen erfolgt mit aufsteigenden Nummern, d.h. ST14 ist der Jumper für Kanal 1 und ST15 für Kanal 2.

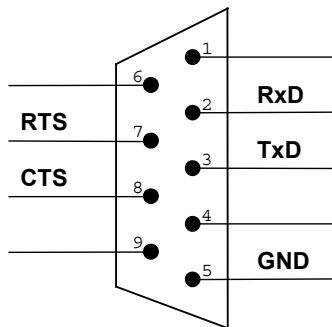
5 Hardwarebeschreibung

5.1 Serielle Schnittstellen A1

Die Schnittstelle A1 ist die Bedienschnittstelle des RTOS-UH. Über diese Schnittstelle werden z.B. die Einschaltmeldung und ggf. Fehlermeldungen ausgegeben. Diese Schnittstelle ist als 5-Draht RS232-Schnittstelle realisiert. Die Startbaudrate beträgt 57600 Baud.

5.1.1 Belegung A1

Die Schnittstelle A1 ist eine 5-Draht Schnittstelle (ST16). Die Belegung ist PC-kompatible ein 9 poliger SubD-Stecker:



5.2 CAN-Bus

Es steht eine galvanisch getrennte CAN-Busschnittstelle zur Verfügung.

5.2.1 Belegung CAN-Bus

Auf ST11 liegt CAN-High, auf ST12 CAN-Low und auf ST13 CAN-Gnd.

5.3 LCD

Der LCD-Anschluß ist für die Ansteuerung eines AMOLED's mit dem Samsung S6E63D6X ausgelegt. Es können aber auch andere Displays angeschlossen werden. Alle Signale sind über Treiber mit Längswiderständen geführt.

5.3.1 Belegung LCD-Anschluß

ST5	PIN	PIN	ST5
GND1	1	2	+5 Volt
	3	4	A0
/RD	5	6	/WR
D0	7	8	D1
D2	9	10	D3
D4	11	12	D5
D6	13	14	D7
/CE	15	16	/Reset
-4,6V	17	18	Inv
Font	19	20	GND1
Y+	21	22	X+
Y-	23	24	X-
/DOFF	25	26	3,3V

5.4 Leistungsausgänge

5.4.1 Solid State Relais

Es stehen 4 SSR zur Verfügung. Jedes SSR ist in der Lage induktive Lasten zu schalten. Die Absicherung erfolgt mit einer flinken 2 A Sicherung. Belegung der SSR:

Funktion	SSR1	SSR2	SSR3	SSR4
SSR Ausgang	ST21	ST24	ST27	ST30
NULL	ST22	ST25	ST28	ST31
PE	ST23	ST26	ST29	ST32

5.4.2 Relais 230V

4 Relaisausgänge zum direkten Schalten von 230V ~ sind verfügbar. Die Relais können wahlweise als mechanische Relais oder SSR bestückt werden. Die Absicherung erfolgt mit einer flinken 2 A Sicherung. Belegung:

Funktion	REL1	REL2	REL3	REL4
Relais Ausgang	ST49	ST52	ST55	ST58
NULL	ST50	ST53	ST56	ST59
PE	ST51	ST54	ST57	ST60

5.4.3 Relais potenzialfrei

2 Relais stellen einen potentialfreien Schließer, 2 Relais einen potenzialfreien Umschalter zur Verfügung:

Funktion	REL5	REL6	REL7	REL8
Relais 11	ST61	ST64	ST67	ST70
Relais 14 (Schließer)	ST62	ST65	ST68	ST71
Relais 12 (Öffner)	-	-	ST69	ST72
PE	ST63	ST66	-	-

5.5 Digitale Eingänge

Die digitalen Eingänge DI1-DI8 sind einseitig auf Masse bezogen, d.h. es kann einfach ein Schalter angeschlossen werden:

Funktion	DI1	DI2	DI3	DI4	DI5	DI6	DI7	DI8
Eingang	ST34	ST36	ST38	ST40	ST42	ST44	ST46	ST48
GND	ST33	ST35	ST37	ST39	ST41	ST43	ST45	ST47

5.6 Analoge Ausgänge

Die analogen Ausgänge sind für 0..20mA ausgelegt. Der Rückkanal der analogen Ausgänge ist jeweils über eine selbstrückstellende 35 mA Sicherung geführt, die bei zu großen Unterschieden im Ground-Potential auslöst. Spannungsausgänge von 0-10 Volt stehen durch Schließen einer Bürde von 536 Ω zur Verfügung.



Funktion	AO1	AO2	Sicherung
Ausgang	ST17	ST19	SI1
GND	ST18	ST20	SI2

5.7 Analoge Eingänge

Die analogen Eingänge sind als differentielle Eingänge ausgeführt. Der Eingangsspannungsbereich kann über eine unterschiedliche Bestückung in weiten Grenzen variiert werden.

Eingang	AI+	AI-	Default Auslegung	Bereich
AI1	ST73	ST74	Pt1000	-30-120°C
AI2	ST75	ST76	Pt1000	-30-120°C
AI3	ST77	ST78	Pt1000	-30-120°C
AI4	ST79	ST80	Pt1000	-30-120°C
AI5	ST81	ST82	Pt1000	-30-120°C
AI6	ST83	ST84	Pt1000	-30-120°C
AI7	ST85	ST86	Pt1000	-30-120°C
AI8	ST87	ST88	Pt1000	-30-120°C
AI9	ST89	ST90	Pt1000	-30-120°C
AI10	ST91	ST92	Pt1000	-30-120°C
AI11	ST93	ST94	Pt1000	-30-120°C
AI12	ST95	ST96	Pt1000	-30-120°C
AI13	ST97	ST98	Pt1000	-30-120°C
AI14	ST99	ST100	Pt1000	-30-120°C
AI15	ST101	ST102	Spannung	0...10 V
AI16	ST103	ST104	Strom	0...20mA

5.8 Leuchtdioden

Die LED LE5 zeigt das Vorhandensein der 3,3 Volt Versorgungsspannung an.

5.9 Batterie/Goldcap

Das Gerät ist ggf. mit einem gepufferten RAM ausgestattet. Die Pufferung erfolgt entweder über eine 3,0 Volt Lithiumbatterie oder einen 1 F Goldcap. Die Batterie wird erst bei der Lieferung eingesetzt und hat dann eine Mindestlebensdauer von 5 Jahren, unabhängig von der Einschaltdauer des Gerätes. Die Pufferdauer mit dem Goldcap hängt vom Ladezustand ab, bei geladenem Goldcap ist eine Pufferdauer von min. 14 Tagen gegeben.

5.10 Socket Modem

Die UCT kann mit unterschiedlichen Socket Modems bestückt werden:

- Analoges V92 Modem
- ISDN Modem
- GSM/GPRS Modem
- Ethernet LAN Modem
- WLAN Modem

Es kann jeweils nur 1 Modem bestückt werden. Die Kommunikation wird über die serielle Schnittstelle A2 durchgeführt.

6 Programmierung

Die Programmierung der auf dem UCT befindlichen Bausteine wird vom Betriebssystem durchgeführt. Die Programmierung aller externen Komponenten wird im Folgenden beschrieben. Absolut notwendig zum Verständnis ist die detaillierte Kenntnis des MC68332 User's Manual.

Es ist bei <http://www.freescale.com> erhältlich. Für die Programmierung der TPU sind die Applikation-Notes hilfreich.

6.1 Adreßbelegung

Chip-Select	Anschluß	Größe	Adresse	Programmierung
CSBOOT	FLASH	bis 1 MB	\$00800000-\$008FFFFFFF	0 WS
CS0	FLASH	bis 2 MB	\$00900000-\$009FFFFFFF	0 WS
CS1	FLASH	bis 3 MB	\$00A00000-\$00AFFFFFFF	0 WS
CS2	FLASH	bis 4 MB	\$00B00000-\$00BFFFFFFF	0 WS
CS3	RAM	bis 1 MB	\$00000000-\$000FFFFFFF	Fast
CS4	RAM	bis 2 MB	\$00100000-\$001FFFFFFF	Fast
CS9	Display	2 KB	\$00D00000-\$00D007FF	Fast

6.2 Interruptquellen

Quelle	Funktion	Level
IRQ1	CAN	1
IRQ2	TouchController	2

Die restlichen IRQ-Pin's werden anderweitig genutzt.

6.2.1 Events

Folgende Events sind belegt:

Event	Funktion	Eingang
EV 00000040	TPU Kanal 6	DI1
EV 00000080	TPU Kanal 7	DI2
EV 00000100	TPU Kanal 8	DI3
EV 00000200	TPU Kanal 9	DI4
EV 00000400	TPU Kanal 10	DI5
EV 00000800	TPU Kanal 11	DI6
EV 00001000	TPU Kanal 12	DI7
EV 00002000	TPU Kanal 13	DI8

Damit der entsprechende Event ausgelöst wird, muß der Interrupt physikalisch freigegeben werden. Default ist ein Event bei jedem Flankenwechsel.

6.3 Watchdog

Der interne Watchdog des MC68332 wird vom Betriebssystem Initialisiert und von der Task Watch auf sehr hoher Priorität (1) im 4s-Zyklus getriggert. Unterbleibt die Triggerung des Watchdogs für mehr als 8 s, wird ein Reset des UCT ausgelöst.

Die Task Watch kann problemlos terminiert und mit einer anderen Wunschkriorität wieder aktiviert werden. Auch eine Neutriggerung des Watchdogs direkt über eine Anwendung ist möglich.

6.4 Ein-/Ausgänge

6.4.1 Digitale Eingänge

Die digitalen Eingänge sind auf TPU-Kanäle geführt und damit vielseitig nutzbar. Beim Start sind alle Eingänge mit der DIO Funktion gestartet. Belegung:

Eingang	TPU
DI1	TPU Kanal 6
DI2	TPU Kanal 7
DI3	TPU Kanal 8
DI4	TPU Kanal 9
DI5	TPU Kanal 10
DI6	TPU Kanal 11
DI7	TPU Kanal 12
DI8	TPU Kanal 13

Die Eingänge sind Interruptfähig und können einen Event auslösen, siehe 6.2.1. Es wird bei jedem Flankenwechsel ein Event ausgelöst, die Eingänge können aber beliebig umprogrammiert werden, so dass ein Event nur bei fallender oder steigender Flanke ausgelöst wird.

6.4.2 Analoge Ein-/Ausgänge, SSR, Relais

Die Ein-/Ausgänge (außer digitale Eingänge) werden von einer Systemtask betreut, da ein Teil über das QSPI angeschlossen ist und nur koordinierte Zugriffe möglich sind. Es wird eine globale Struktur zur Verfügung gestellt, die den weiter unten beschriebenen Aufbau hat. Zuerst muß die Adresse der Struktur ermittelt werden. Dazu wird ein READ-Befehl an die Systemtask geschickt:

```

DAT: LD/5.8/UCT(NE)<->;
...
DCL p_uct REF UCTBUF ;
DCL ptr STRUCT [ p_uct REF UCTBUF ] ;
...
init_io : PROC RETURNS( FIXED(15) ) ;
DCL erg FIXED(15) ;
READ ptr FROM DAT ; /* Pointer muss in Struktur versteckt werden */
erg = 0 ;
erg = ST( DAT ) ;
IF erg EQ 0 THEN
    p_uct = ptr.p_uct ; /* Pointer setzen */
FIN ;
RETURN( erg ) ; /* Bei 0 kann der Pointer genutzt werden */
END;

```

Die Struktur hat folgenden Aufbau:

```

TYPE UCTBUF STRUCT
[
    ad_werte_akt(16) FIXED(15) , /* aktuell gewandelte Werte          */
    ad_werte_mit(16) FIXED(31) , /* Mittelwerte * 10              */
    anz_mittelungen  FIXED(15) , /* Anzahl der Mittelwerte (Init=20) */
    da_werte(2)      FIXED(15) , /* 0-1023 für D/A-Wandler (Init= 0) */
    do_out           BIT(16)   , /* xxxx SSR4 SSR3 SSR2 SSR1
                                REL8 REL7 REL6 REL5
                                REL4 REL3 REL2 REL1          */
    sample_all       FIXED(15) , /* =0 immer nur einen Kanal wandeln
                                !=0 alle Kanäle wandeln          */
    sample_time      FIXED(15) , /* Resume in ms                  (Init=5) */
    tid              BIT(32)   /* Betreuungstask Grundebene     */
] ;

```

Nach dem erfolgreichen Aufruf von `init_io` kann über z.B. `p_uct.ad_werte_mit(2)` auf den Mittelwert des 2.ten AD-Kanals zugegriffen werden.

6.4.2.1 Beschreibung der Strukturelemente

`ad_werte_akt(i)` Hier stehen die aktuellen Werte der 16 analogen Eingänge. Der Wertebereich ist 0-4095 (12 Bit).

`ad_werte_mit(i)` enthält die Mittelwerte der analogen Eingänge. Die Mittelwerte sind mit dem Faktor 10 skaliert, d.h. der Wertebereich ist 0-40950.

anz_mittelungen legt die Anzahl der Werte, die für die Mittellung herangezogen werden, fest. Maximal darf ein Wert von 32000 eingetragen werden.

da_werte(i) enthält den Ausgangswert der beiden analogen Ausgangs-Kanäle. Der Wertebereich beträgt 0-1023.

do_out legt die 16 digitalen Ausgänge fest. Dabei gilt folgende Zuordnung:

x	x	x	x	SSR4	SSR3	SSR2	SSR1
REL8	REL7	REL6	REL5	REL4	REL3	REL2	REL1

Mit einer '1' wird der Ausgang eingeschalter, mit '0' aus.

sample_all legt fest, ob bei jedem Durchlauf der Betreuungstask alle A/D-Kanäle oder nur einer gewandelt wird. Bei einem Wert von 0 wird immer nur ein Kanal gewandelt, d.h. es werden 16 Durchläufe benötigt um alle Kanäle zu aktualisieren. Bei allen anderen Werten werden alle 16 A/D-Kanäle gewandelt.

sample_time bestimmt die Länge der Pause in ms, bis die Betreuungstask wieder aktiv wird. Das System startet mit 5 ms, wird die Zeit kleiner gewählt, steigt die Rechnerauslastung.

6.5 Serielle Schnittstelle

Für die serielle Schnittstelle finden folgende Port-Anschlüsse Verwendung:

Signal	Port
TxD_1	TXD1
RxD_1	RXD1
RTS_1	TPU1
CTS_1	TPU0

6.6 Socket Modem

Das Socket Modem kann über die MODCK-Leitung des Prozessors einen Reset bekommen. Dazu muß MODCK als Ausgang konfiguriert werden, dies führt das Betriebssystem durch. Ins Port F Data Direction Register (0xFFFA1D) muß statt 0xF8 eine 0xF9 eingetragen werden:

```
sm-b FFFA1D F9
```

Nun kann im Port F Data Register (\$FFFA19) das unterste Bit auf 0 und dann wieder auf 1 gesetzt werden, damit wird ein Reset des Socket Modems ausgelöst. Bitte beachten Sie, dass die restlichen Bits nicht verändert werden dürfen, an den oberen 4 Bit sind z.B. die Solid State Relais angeschlossen!



7 Update RTOS-UH

Um das RTOS-UH Betriebssystem upzudaten sind die folgenden Schritte notwendig. Bitte beachten Sie, dass ein Fehler oder Spannungsausfall zur kompletten Unbrauchbarkeit der UCT führen kann!



1. Beenden Sie **alle** Programme auf der UCT! Der Bereich von 0x1FFF0 bis 0xB0000 muß frei sein.
2. Laden Sie den RTOS-UH S-Record auf die Adresse 0x1FFF0.
3. Starten Sie das neue RTOS-UH mit dem Befehl "go a000e".
4. Das neue System muß starten.
5. Mit dem Befehl "rtboot -a 6 -d -h 20000 -e 7ffff -u" wird das neue System ins Flash gebrannt.
6. Nun muß sich nach einem Restart das neue System aus dem Flash melden.

7.1 FLASH-Belegung

Das FLASH ist folgendermassen aufgeteilt:

Adresse	Funktion
0x800000-0x82FFFF	RTOS-UH
0x830000-0x83FFFF	RT-FLASH
0x840000-0x85FFFF	Terminalemulation
0x860000-0x87FFFF	Reserved
0x880000-0xBFFFFFFF	Free

7.2 RTOS-UH ohne Anwender-Flash starten

Ab Rev. 1.1 ist auf der Platine der Jumper JU1 (oberhalb des SocketModems) vorhanden. Wird dieser beim Einschalten geschlossen, so startet RTOS-UH ohne Anwendungsprogramme aus dem FLASH zu starten. Damit kann ein evt. "verstorbene" System mit einem fehlerhaften Anwendungsprogramm wieder zum Leben erweckt werden.

8 Terminalemulation

8.1 Allgemeines

Die Terminalemulation ermöglicht den Anschluß von getrennten Tastaturen und Displays an einem RTOS-UH-Rechner und verhält sich wie eine übliche serielle Schnittstelle.

Unter den mnemotechnischen Bezeichnungen /AT, /BT und /CT ist die Terminalemulation in der A-, B- oder C-Betriebsart unter der LDN 4 mit den Drives 0, 2 und 6 erreichbar. Unter der Bezeichnung /DT wird der Ausgabekanal im Vollduplex-Betrieb mit der LDN 14 angesprochen.

Zusätzlich zu den üblichen Betriebsarten kann die Terminalemulation bei LDN 4 noch unter den Drives 64, 66 und 70 angesprochen werden. Die Terminalemulation liefert in diesem Fall bei Eingaben die Tastatur-Scancodes ohne Übersetzung in ASCII-Zeichen. Auch hier ist ein Betrieb entsprechend der A-, B- oder C-Betriebsart möglich, jedoch findet grundsätzlich kein Echo statt. Besonders ist zu beachten:

- Bei Betriebsartenumschaltung Scan-Code/ASCII kann der Empfangspuffer noch Zeichen in der jeweils anderen Codierung enthalten. Zur Sicherheit sollte der Empfangspuffer daher durch ein erstes Lesen in der A-Betriebsart gelöscht werden.
- Bei Scan-Code-Betrieb müssen die Ausgaben ebenfalls über die Drives 64, 66 oder 70 erfolgen, da ansonsten wieder eine Rückschaltung in den ASCII-Betrieb erfolgt.

Der Tastaturreiber bietet keine Auto-Repeat-Funktion. Statt dessen sind Treiberversionen verfügbar, die bei Loslassen einer gedrückten Taste den ASCII-Wert \$FF liefern. Hierdurch kann die Dauer einer Tastenbetätigung erfasst und ggf. ein Auto-Repeat emuliert werden.

8.2 Befehlsvorrat Terminalemulation

Die Terminalemulation verhält sich weitgehend Televideo-kompatibel. Sie versteht die folgende Befehlssequenzen (Erläuterung der Angaben *Ps*, *Pc*, *Pn*, *r* und *c* siehe unten). Es ist nicht gewährleistet, daß alle Funktionen auf allen Terminals zur Verfügung stehen! So können auf einen schwarz/weiß-Display natürlich keine Farben eingestellt werden, oder auf einen Textdisplay wird es schwierig, Grafik auszugeben.

Beachten Sie bitte, dass die folgenden Sequenzen nur nutzbar sind, wenn der Grafik-Mode nicht benutzt wird!



Cursor style (0..4)		
	ESC . <i>Ps</i>	
	0	Cursor off
	1	Blinking block
	2	Steady block
	3	Blinking underline
	4	Steady underline

Define Visual attributes																			
	ESC	G	P	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
			Underline									X	X	X	X	X	X	X	X
			Reverse					X	X	X	X					X	X	X	X
			Blinking			X	X			X	X			X	X			X	X
			Invisible		X		X		X		X		X		X		X		X
Operating Modes																			
)	Enable invers															
			(Disable invers															
			V	Autoscroll off															
			W	Autoscroll on															
			\$	Grafics mode on															
			%	Grafics mode off															
			U	Monitor mode on															
			X	Monitor mode off															
			[=7h	Auto Wrap on															
			[=7l	Auto Wrap off															
			! c	c >= ,A': Schreibfarbe, c<'A': Hintergrundfarbe setzen															
			[Pc ; Pc r	Set scroll region rows (Pc: start, end)															
			[Pc ; Pc s	Set scroll region columns (Pc: start, end)															
Cursor Movement																			
			J	Line feed															
			K	Up															
			V	Down															
			L	Right															
			M	Carriage Return															
			H	Left															
			= r c	Set Cursor row column															
			[Pn A	Cursor up Pn rows															
			[Pn B	Cursor down Pn rows															
			[Pn C	Cursor right Pn columns															
			[Pn D	Cursor left Pn columns															
			[c ; r H	Set Cursor position column, row															
Tabulator Control																			
			1	Set tab stop at actual column															
			2	Clear tab stop at actual column															
			3	Clear all tab stops															
			i	Move Cursor to next tab stop															
			l	Move Cursor back one tab stop															
Insertion																			
			Q	Insert space at cursor															
			E	Insert line of spaces															
Miscellaneous																			
			~	Reset Terminal															
Deletion																			
			W	Delete char at cursor															
			R	Delete current line to spaces															
			T	Delete to end of line															

	t	
	Y	Delete to end of screen
	y	
	*	Delete Screen
	,	
	;	
	+	
	:	
CTRL Z		

Die Befehlssequenz zur Änderung der Schreibfarbe (ESC ! c) ist nicht zu gängigen Terminal emulationen kompatibel. Neben der einfachen Änderung der Schreibfarbe kann durch Änderung des Grundwertes der Farbangabe folgendes Verhalten erzeugt werden:

Grundwert	Verhalten
32 (Leerzeichen)	Änderung der Schreibfarbe
64 (A)	Änderung der Hintergrundfarbe

Die Angaben für mit *Ps*, *Pc*, *Pn*, *r* oder *c* gekennzeichnete Zahlenwerte erfolgen durch Werte, die sich ASCII-codiert aus 32 + Zahlenwert errechnen.

8.3 Umsetzung der Eingabezeichen

Bei der Matrixtastatur sind unterschiedliche Belegungen möglich. So können z.B. sämtliche Tasten als frei programmierbare Funktionstasten zur Verfügung gestellt werden. Bei der PC-Tastatur sind die normalen Funktionstasten belegt, die Tasten von 13 bis 24 werden durch gleichzeitiges Drücken von *Shift* und *Funktionstaste* erreicht. Standardmäßig liefern die Funktionstasten *S0H zeichen CR* mit folgendem *zeichen*

Funktionstaste	Zeichen	Shift+F-Taste	Zeichen
1	@	13	'
2	A	14	a
3	B	15	b
4	C	16	c
5	D	17	d
6	E	18	e
7	F	19	f
8	G	20	g
9	H	21	h
10	I	22	i
11	J	23	j
12	K	24	k

Bei Matrixtastaturen hängt die Zuordnung der einzelnen Tasten zu den Tastencodes von dem Anschluß der Tastatur ab und muß für jeden Anwendungsfall ermittelt werden.

Die Programmierung der Funktionstasten erfolgt mit der Befehlssequenz

ESC | p11 text CTRL-Y

und den Parametern

p1 Kennzeichnung der zu programmierenden Taste

text gewünschte Tastenbelegung

Funktionstaste	Kennung	Shift+F-Taste	Kennung
1	1	13	<
2	2	14	=
3	3	15	>
4	4	16	?
5	5	17	@
6	6	18	A
7	7	19	B
8	8	20	C
9	9	21	D
10	:	22	E
11	;	23	F
12	G	24	L

8.4 Grafikfunktionen

Bei grafikfähigen Displays stellt die Terminalemulation Funktionen zur Nutzung der grafischen Fähigkeiten des Displays zur Verfügung. Der Leistungsumfang richtet sich nach der Leistungsfähigkeit des jeweiligen Grafik-Displays bzw. des verwendeten Controllers.

Die Grafikfunktionen sind aus Geschwindigkeitsgründen nicht auf Multi-Tasking ausgelegt. Grundsätzlich sollte ein Grafiksystem nur von einer Task angesprochen werden. Besonders der kombinierte Betrieb von Terminalemulation und Grafikfunktionen erfordert besondere Aufmerksamkeit.

Bitte beachten Sie, dass bei der Nutzung des Grafik-Mode bei bestimmten Displays der Attributspeicher nicht mehr zur Verfügung steht. Dies hat z.B. zur Folge, dass die Ausgabe von invertiertem Text nicht mehr möglich ist.



Neben den Grafik-Primitiven SETPIX, GETPIX und LINE (s. RTOS-UH-Handbuch Kapitel 5.7.3) gibt es folgende Prozeduren:

8.4.1 Funktionsumfang und Implementierungsabhängigkeiten

Prozedur	IP-SVGA	PVGA	VHI	NBS300	UCT	MPC555
BACK PEN	Ja	Ja	—	—	Ja	—
BIN TEXT	Ja	Ja	Ja	Ja	Ja	Ja
BOX	Ja	Ja	Ja	Ja	Ja	Ja
BOX FILLED	Ja	Ja	Ja	Ja	Ja	Ja
BOX MOVE	Ja	Ja	Ja	Ja	Ja	Ja
BOX READ	Ja	Ja	—	—	Ja	—
BOX WRITE	Ja	Ja	—	—	Ja	—
CIRCLE	Ja	Ja	Ja	Ja	Ja	Ja
CLEAR	Ja	Ja	Ja	Ja	Ja	Ja
COLOR CLEAR	Ja	Ja	—	—	Ja	—
DISPLAY MODE	Ja	Ja	Ja	Ja	—	Ja
DISPLAY STATE	Ja	Ja	—	—	Ja	—
DOT LINE	Ja	Ja	Ja	Ja	Ja	Ja
DOT LINE RANGE	Ja	Ja	Ja	Ja	Ja	Ja
FILLED HISTOGRAM	Ja	Ja	Ja	Ja	Ja	Ja
GET PEN	Ja	Ja	—	—	Ja	—
GET BACK PEN	Ja	Ja	—	—	Ja	—
GET TEXT FONT	Ja	Ja	—	—	Ja	—
GET TEXT WIDTH	Ja	Ja	Ja	Ja	Ja	Ja
GRAPH CURSOR	Ja	Ja	—	—	—	—
GRAPH CURSOR SELECT	Ja	Ja	—	—	—	—
HIDE CURSOR	Ja	Ja	—	—	Ja	—
HISTOGRAM	Ja	Ja	Ja	Ja	Ja	Ja
PEN	Ja	Ja	—	—	Ja	—
PLANE	—	Ja	—	—	—	—
PLINIT	Ja	Ja	Ja	Ja	Ja	Ja
POLYGON	Ja	Ja	Ja	Ja	Ja	Ja
SELECT FONT	Ja	Ja	—	—	Ja	—
SELECT TEXT FONT	Ja	Ja	—	—	Ja	—
SET TEXT WIDTH	Ja	Ja	Ja	Ja	—	Ja
SET VGA SCREEN	Ja	Ja	—	—	—	—
TEXT	Ja	Ja	Ja	Ja	Ja	Ja
VGA GET PLANE	Ja	Ja	—	—	—	—
VGA LOCK TEXT	Ja	Ja	—	—	—	—
VGA SET PLANE	Ja	Ja	—	—	—	—
VGA UNLOCK TEXT	Ja	Ja	—	—	—	—
WIDTH	Ja	Ja	Ja	Ja	Ja	Ja

Das UCT unterstützt nur Zeichensätze von \$20 bis \$7F.

8.4.2 Allgemeines

Zum Verständnis der Funktionsbeschreibung sind Kenntnisse über die folgenden Begriffe erforderlich:

➤ Farbe

Farben werden in RGB-Notation angegeben. Die Angabe kennzeichnet die Intensität, in der der jeweilige Farbanteil dargestellt wird:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	R	R	R	R	R	G	G	G	G	G	B	B	B	B	B

Es ergeben sich also folgende Muster, um die reine Grundfarbe zu Zeichnen:

Rot: 0x7C00

Grün: 0x03E0

Blau: 0x001F

Das oberste Bit legt den Zeichenmode fest, mit 0 wird absolut gezeichnet, mit 1 im XOR-Mode.

➤ Pixel

Kennzeichnet einen Bildpunkt. Je nach verwendetem Grafiksystem kann ein Bildpunkt unterschiedliche Farben annehmen.

➤ Pen

Ein Pen ist durch eine Farbe gekennzeichnet.

Es gibt einen Schreib- und einen Hintergrund-Pen. Pixel, Linien u.ä. werden nur unter Verwendung des Schreib-Pens gezeichnet.

Vorgegebene Konstrukte wie Fonts o.ä. füllen den Zeichenhintergrund mit dem Hintergrund-Pen und stellen die Zeichen mit dem Schreib-Pen dar.

➤ Font

Ein Font (Zeichensatz) ist als Bitmap abgelegt. Alle Zeichen eines Fonts haben gleiche Größe. Die Größe eines Fonts beinhaltet Ober- und Unterlängen.

Die Bitmap enthält Informationen darüber, welche Pixel der von einem Zeichen eingenommen Fläche mit dem Schreib-Pen und welche Pixel mit dem Hintergrund-Pen darzustellen sind. Ein Font selber enthält keine Farbinformationen.

Auf dem UCT stehen folgende Fonts zur Verfügung:

Fontnummer	Größe	Zeichengröße
0	8	8x12
1	10	11x14
2	12	14x18
3	14	16x22
4	18	20x26

➤ Palette oder Plane

Die im Bildschirmspeicher abgelegten Pixel-Farbinformationen werden über Paletten in tatsächliche Farbtintensitätswerte übersetzt.

Die Pixel-Farbinformation kann in diesem Fall mit geringem Speicherbedarf abgelegt werden (z.B. 4 Bit). Diese Information wird als Index beim Zugriff auf die Palette verwendet. Die Palette liefert dann die tatsächliche Farbinformation als RGB-Wert mit deutlich höherer Farbaufösung (z.B. 3x 8 Bit).

➤ Screen

Ein Screen ist die Menge aller für einen sichtbaren Bildschirm im Bildspeicher abgelegten Daten. Bei Grafiksystemen, die entweder mehrere unterschiedliche Bildschirme unterstützen oder im Bildspeicher gleichzeitig Daten für mehrere Bildschirme ablegen können, ist die Wahl des Screens, auf dem die folgenden Zeichenoperationen erfolgen sollen, möglich.

8.4.3 Funktionsreferenz

PEARL-Spezifikation	Funktion
BACK_PEN: PROC(col FIXED(15)) GLOBAL;	Setzt die angegebene Farbe <code>col</code> als Hintergrundfarbe. Bei folgenden Operationen, die eine Hintergrundfarbe darstellen müssen, aber keine eigene Hintergrundfarbe spezifizieren (<code>TEXT</code> und <code>BIN_TEXT</code>), wird <code>col</code> als Hintergrundfarbe verwendet.
BIN_TEXT: PROC((x, y) FIXED(15), col FIXED(15), zeichen_adr FIXED(31)) GLOBAL;	Überträgt ein Zeichen aus dem Zeichengenerator-ROM in der Farbe <code>col</code> auf den Bildschirm an die Position (x, y). Die Adresse im ROM wird mit <code>zeichen_adr</code> angegeben. Sie hängt von der Größe des auszugebenden Zeichens ab: $\text{zeichen_adr} = \frac{\text{xhöhe} * \text{ybreite} + 7}{8}$
BOX: PROC((x, y) FIXED(15), (Xend, Yend) FIXED(15), col FIXED(15)) GLOBAL;	Zeichnet einen rechteckigen Rahmen in der Farbe <code>col</code> mit den diagonalen Punkten (x, y) (linke obere Ecke) und (Xend, Yend) (rechte, untere Ecke).
BOX_FILLED: PROC((x, y) FIXED(15), (Xend, Yend) FIXED(15), col FIXED(15)) GLOBAL;	Zeichnet ein in der Farbe <code>col</code> gefülltes Rechteck mit den diagonalen Punkten (x, y) und (Xend, Yend)
BOX_MOVE: PROC((Xstart, Ystart) FIXED(15), (Breite, Hoehe) FIXED(15), (Xziel, Yziel) FIXED(15), mode FIXED(15)) GLOBAL;	Kopiert einen rechteckigen Bildausschnitt von Breite und Höhe beginnend bei dem Startpunkt (Xstart, Ystart) auf einen Bildbereich gleicher Größe mit dem Startpunkt (Xziel, Yziel). Der Zielbereich kann außerhalb des sichtbaren Bildschirmbereichs liegen, d.h. Ywidth wird überschritten. mode legt die Zeichenart fest, im unteren Nibble sind folgende Werte zulässig:

	<p>Dez. hex Art</p> <p>12 \$0C absolut 3 \$03 not 8 \$08 and 14 \$0E or 6 \$06 exor</p> <p>Das nächste Nibble legt die Schreibfarbe fest, d.h. für ein Pixel werden die 4 Bit aus dem Video-RAM entsprechend dem unteren Nibble gelesen, mit der Schreibfarbe „verundet“ und wieder im Video-RAM abgelegt. Wird der Parameter <code>mode</code> auf absolut (=12) gesetzt, so wird der original Bildausschnitt an der Zielstelle abgelegt.</p>
<pre>BOX_READ: PROC((Xstart, Ystart) FIXED(15), (Breite, Hoehe) FIXED(15), feld STRUCT[REF CHAR(1)]) GLOBAL;</pre>	<p>Kopiert einen rechteckigen Bildausschnitt mit <code>Breite</code> und <code>Hoehe</code> beginnend bei dem Startpunkt (<code>Xstart</code>, <code>Ystart</code>) in den Speicherbereich, der mit <code>feld</code> angegeben wird. Dabei findet keine Überprüfung der Größe des Speicherbereiches statt.</p> <p>Ist das angegebene <code>feld</code> zu klein, ist der Absturz des Rechners so gut wie sicher!</p>
<pre>BOX_WRITE: PROC((Xstart, Ystart) FIXED(15), (Breite, Hoehe) FIXED(15), mode FIXED(15), feld STRUCT[REF CHAR(1)]) GLOBAL;</pre>	<p>Kopiert den Speicherbereich, der mit <code>feld</code> angegeben wird, auf einen rechteckigen Bildausschnitt von <code>Breite</code> und <code>Höhe</code>, beginnend bei dem Startpunkt (<code>Xstart</code>, <code>Ystart</code>).</p> <p>Dabei findet keine Überprüfung der Größe des Speicherbereiches statt. Ist das angegebene <code>feld</code> zu klein, wird der folgende Speicher auf den Bildschirm geschrieben.</p> <p>Der Parameter <code>mode</code> ist bei der Prozedur <code>BOX_MOVE</code> beschrieben.</p>
<pre>CIRCLE: PROC((Xmitte, Ymitte) FIXED(15), Radius FIXED(15), col FIXED(15)) GLOBAL;</pre>	<p>Zeichnet geschlossene Kreislinien auf den Bildschirm.</p>
<pre>CLEAR: PROC GLOBAL;</pre>	<p>Löscht den sichtbaren Bildschirmbereich mit der Farbe 0.</p>
<pre>COLOR_CLEAR: PROC(col FIXED(15)) GLOBAL;</pre>	<p>Löscht den sichtbaren Bildschirmbereich mit der angegebenen Farbe <code>col</code>.</p>
<pre>DISPLAY_MODE: PROC(mode FIXED(31)) GLOBAL;</pre>	<p>Ermöglicht bei einigen Systemen die Umschaltung der Betriebsart des Grafiksystems (Text=0/Grafik=1).</p>
<pre>DISPLAY_STATE: PROC(State FIXED(31)) GLOBAL;</pre>	<p>Schaltet Display ein (<code>State = 1</code>) oder aus (<code>State = 0</code>)</p>
<pre>DOT_LINE: PROC((Xstart, Ystart) FIXED(15), (XEnd, YEnd) FIXED(15), col FIXED(15), mask FIXED(15)) GLOBAL;</pre>	<p>Zeichnet ein (strichpunktierte) Linie vom Punkt (<code>Xstart</code>, <code>Ystart</code>) zum Punkt (<code>XEnd</code>, <code>YEnd</code>) in der Farbe <code>col</code>.</p> <p>Mit <code>mask</code> wird das Linienmuster vorgegeben: -1: durchgezogene Linie</p>

) GLOBAL;	TOFIXED 'AAAA'B4: einfach punktiert
DOT_LINE_RANGE: PROC((Xstart, Ystart) FIXED(15), (XEnd, YEnd) FIXED(15), col FIXED(15), mask FIXED(15), (YU, YO) FIXED(15), col2 FIXED(15),) GLOBAL;	Zeichnet ein (strichpunktierte) Linie vom Punkt (Xstart, Ystart) zum Punkt (XEnd, YEnd) in der Farbe col. Mit mask wird das Linienmuster vorgegeben: -1: durchgezogene Linie TOFIXED 'AAAA'B4: einfach punktiert Die Y-Koordinaten YU (unten) und YO (oben) sind Umschaltsschwellen für die Zeichenfarbe. Liegt ein zu zeichnender Bildpunkt unterhalb YU oder oberhalb YO, so wird er in der Farbe col2 anstatt in col dargestellt.
FILLED_HISTOGRAMM: PROC((XS, YS) FIXED(15) IDENT, count FIXED(15), col FIXED(15), b_col FIXED(15), width FIXED(15), ySave FIXED(15) IDENT) GLOBAL;	Zeichnet ein Histogramm aus count nicht gefüllten Balken der Breite width in der Farbe col. Die einzelnen Histogrammbalken werden durch ihre linke, obere Ecke (XS, YS) und die Breite width bestimmt. XS und YS müssen in getrennten FIXED(15)-Feldern gegeben werden. Die Y-Grundlinie des Histogramms wird durch die Initialwerte im Feld ySave gegeben. Zur Erhöhung der Zeichengeschwindigkeit bei wiederholter Darstellung eines Histogramms wird nur die Änderung der Balkenhöhe gegenüber der im Feld ySave gespeicherten Höhe neu gezeichnet.
GET_PEN: PROC RETURNS(FIXED(15)) GLOBAL;	Gibt die Farbe des aktuell gewählten Schreib-Pens zurück.
GET_BACK_PEN: PROC RETURNS(FIXED(15)) GLOBAL;	Gibt die Farbe des aktuell gewählten Hintergrund-Pens zurück.
GET_TEXT_FONT: PROC RETURNS(FIXED(15)) GLOBAL;	Gibt die Nummer des aktuellen Textfonts zurück (von SELECT_FONT oder SELECT_TEXT_FONT gesetzt). Anzahl und Art der verfügbaren Fonts sind abhängig vom eingesetzten Grafiksystem.
GET_TEXT_WIDTH: PROC(Xhoehe FIXED(15) IDENT, ybreite FIXED(15) IDENT) GLOBAL;	Liefert die Größe eines Zeichens des gewählten Zeichensatzes in den Variablen xhoehe und ybreite zurück.
GRAPH_CURSOR: PROC((x, y) FIXED(15), status FIXED(15)) GLOBAL;	Positioniert den Graphik-Cursor auf den Punkt (x, y). Mit status = 1 wird der Graphik-Cursor sichtbar, mit status = 0 wird er unsichtbar.
GRAPH_CURSOR_SELECT: PROC(Cursor FIXED(31) IDENT, mode FIXED(15)) GLOBAL;	Gibt eine Bitmap zur Darstellungsform des Graphik-Cursors vor. Cursor muß das erste Element eines FIXED(31)-Feldes mit 32 Elementen sein; damit wird ein 32x32 Bit-Cursor beschrieben. Gesetzte Bits entsprechen darzustellenden Pixeln. mode gibt die Darstellungsart des Cursors an.
HIDE_CURSOR: PROC GLOBAL;	Schaltet den Graphik-Cursor aus.
HISTOGRAMM: PROC((XS, YS) FIXED(15) IDENT, count FIXED(15), col FIXED(15),	Zeichnet ein Histogramm aus count nicht gefüllten Balken der Breite width in der Farbe col. Die Y-Grundlinie des Histogramms wird in ofs gege-

<pre> b_col FIXED(15), (off_s, width) FIXED(15), ySave FIXED(15) IDENT) GLOBAL;</pre>	<p>ben.</p> <p>Die einzelnen Histogrammbalken werden durch ihre linke, obere Ecke (XS, YS) und die Breite width bestimmt.</p> <p>XS und YS müssen in getrennten FIXED(15)-Feldern gegeben werden.</p> <p>Zur Erhöhung der Zeichengeschwindigkeit bei wiederholter Darstellung eines Histogramms wird nur die Änderung der Balkenhöhe gegenüber der im Feld ySave gespeicherten Höhe neu gezeichnet. Bei der erstmaligen Darstellung eines Histogramms muss ySave daher vollständig mit dem Wert off_s initialisiert werden.</p>												
<pre> PEN: PROC(col FIXED(15)) GLOBAL;</pre>	<p>Setzt die angegebene Farbe col als Schreibfarbe. Bei folgenden Operationen, die keine eigene HinSchreibfarbe spezifizieren, wird col als Schreibfarbe verwendet.</p>												
<pre> PLANE: PROC(Farbnummer FIXED(15), Palettenwert FIXED(31)) GLOBAL;</pre>	<p>Setzt den Wert eines Paletteneintrags. Für Farbnummer sind Werte von 0 bis 15 zulässig. Folgende Bits vom Palettenwert werden genutzt (abhängig vom Grafiksyste(m)):</p> <table border="1" data-bbox="675 1025 1193 1160"> <thead> <tr> <th>Farbe</th> <th>Bit (PEARL)</th> <th>Bit(Assembler)</th> </tr> </thead> <tbody> <tr> <td>Rot</td> <td>12 ... 4</td> <td>18 ... 20</td> </tr> <tr> <td>Grün</td> <td>20 ...22</td> <td>10 ... 12</td> </tr> <tr> <td>Blau</td> <td>28 ...30</td> <td>2 ... 4</td> </tr> </tbody> </table> <p>Der Aufbau des Langwortes Palettenwert läßt sich durch die Einzelbit-Darstellung [---- ---- ---r rr-- ---g gg-- ---b bb--] veranschaulichen.</p>	Farbe	Bit (PEARL)	Bit(Assembler)	Rot	12 ... 4	18 ... 20	Grün	20 ...22	10 ... 12	Blau	28 ...30	2 ... 4
Farbe	Bit (PEARL)	Bit(Assembler)											
Rot	12 ... 4	18 ... 20											
Grün	20 ...22	10 ... 12											
Blau	28 ...30	2 ... 4											
<pre> PLINIT: PROC GLOBAL;</pre>	<p>Initialisiert den Display-Prozessor. Muß vor Benutzung der Grafik-Routinen einmal aufgerufen werden.</p>												
<pre> POLYGON: PROC((x, y) FIXED(15) IDENT, cnt FIXED(15), col FIXED(15)) GLOBAL;</pre>	<p>Zeichnet einen (nicht geschlossenen) Linienzug mit cnt Stützpunkten in der Farbe col.</p> <p>X und y müssen in getrennten, eindimensionalen Feldern abgelegt sein.</p>												
<pre> SELECT_FONT: PROC(Font FIXED(15)) GLOBAL;</pre>	<p>Wählt den angegebenen Font als aktuellen Textfont aus. Anzahl und Art der verfügbaren Fonts sind abhängig vom eingesetzten Grafiksyste(m).</p>												
<pre> SELECT_TEXT_FONT: PROC(Font FIXED(15)) GLOBAL;</pre>	<p>Wählt den angegebenen Font als aktuellen Textfont aus. Anzahl und Art der verfügbaren Fonts sind abhängig vom eingesetzten Grafiksyste(m).</p>												
<pre> SET_TEXT_WIDTH: PROC(Xhöhe FIXED(15) IDENT, ybreite FIXED(15) IDENT) GLOBAL;</pre>	<p>Setzt die Größe eines Zeichens für die Funktionen BIN_TEXT und TEXT.</p>												
<pre> SET_VGA_SCREEN: PROC(Screen FIXED(15)) RETURNS(FIXED(15)) GLOBAL;</pre>	<p>Wählt den angegebenen Screen als Ziel der folgenden Graphikoperationen aus.</p> <p>Der Rückgabewert ist -1, falls der angegebene Screen im aktuellen Grafiksyste(m) nicht vorhanden ist.</p>												

<pre>TEXT: PROC((x, y) FIXED(15), col FIXED(15), string STRUCT [str CHAR(255), len FIXED(15)]) GLOBAL;</pre>	<p>Schreibt die in <code>string.str</code> abgelegte Zeichenkette beginnend auf der Position <code>(x, y)</code> (linke, obere Ecke des ersten Zeichens) in der Farbe <code>col</code> auf den Bildschirm. Die Ausgabe endet, wenn <code>string.len</code> Zeichen ausgegeben wurden. Die Schreibrichtung ist horizontal (x-Richtung).</p>
<pre>VGA_GET_PLANE: PROC(Plane FIXED(15), (R, G, B) FIXED(15) IDENT) GLOBAL;</pre>	<p>Liest aus der Palette <code>Plane</code> den enthaltenen RGB-Wert aus.</p>
<pre>VGA_LOCK_TEXT: PROC RETURNS(FIXED(15)) GLOBAL;</pre>	<p>Sperrt das Grafiksystem gegenüber den Aktionen der Terminalemulation. Diese Funktion muß jeweils vor einem Block von zusammenhängenden Grafikoperationen aufgerufen werden, falls gleichzeitig die Terminalemulation genutzt werden soll. Der Rückgabewert ist 1, falls die Operation erfolgreich war.</p>
<pre>VGA_SET_PLANE: PROC(Plane FIXED(15), (R, G, B) FIXED(15)) GLOBAL;</pre>	<p>Setzt die Palette <code>Plane</code> auf einen angegebenen RGB-Wert. <code>Plane</code> wird beim Zeichnen über die <code>col</code> angesprochen</p>
<pre>VGA_UNLOCK_TEXT: PROC RETURNS(FIXED(15)) GLOBAL;</pre>	<p>Gibt das Grafiksystem nach einem Aufruf von <code>VGA_LOCK_TEXT</code> wieder für Aktivitäten der Terminalemulation frei. Der Rückgabewert ist 1, falls die Operation erfolgreich war.</p>
<pre>WIDTH: PROC(Xwidth FIXED(15) IDENT, Ywidth FIXED(15) IDENT) GLOBAL;</pre>	<p>Gibt in den Variablen <code>Xwidth</code> und <code>Ywidth</code> die Größe des Bildschirms in Pixel zurück.</p>

8.4.4 Zeichensatz

Der Zeichensatz für die Funktion TEXT enthält folgende Zeichen oder den eingebauten Zeichensatz des Displays (es müssen nicht alle Zeichen vorhanden sein!):

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	☺	☹	♥	♦	♣	♠	●	■	○	◼	♂	♀	♪	♫	*
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
▶	◀	↕	!!	¶	§	■	↕	↑	↓	→	←	⊥	↔	▲	▼
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
p	q	r	s	t	u	v	w	x	y	z	{		}	~	△
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
	ú	é	ξ	ä	à	á	ç	ê	ë	è	ï	î	ì	Ä	Å
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	φ	£	¥	Pts	f
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
á	é	í	ú	ñ	Ñ			¿	Γ	γ	½	¼	i	«	»
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
☼	☽	☿		†	‡	¶	π	τ	‡		π	∟	∟	∟	γ
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
L	⊥	⊥	†	—	†	†		ℒ	℞	⊥	π		=		⊥
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
⊥	⊥	π	ℒ	ℒ	℞	π		≠	∟	Γ	■	■	■	■	■
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
α	β	Γ	π	Σ	σ	μ	T	Φ	Θ	Ω	δ	θ	ø	€	Π
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
≡	±	≥	≤			÷	≈	°	▪	·	√	n	²	■	