Einführung in die Software des Gerätesystems P8000

EAW *electronic*

P8000

transcribed in ~22h by O. Lehmann during 2006-07-25 and 2007-06-30
Version 1.9 (2008-07-05)

Einführung in die Software des Gerätesystems P8000

Diese Dokumentation wurde von einem Kollektiv des

Kombinat
VEB ELEKTRO-APPARATE-WERKE
BERLIN-TREPTOW "FRIEDRICH EBERT"

erarbeitet.

Im Interesse einer ständigen Weiterentwicklung werden die Nutzer gebeten, dem Herausgeber Hinweise zur Verbesserung mitzuteilen.

Herausgeber:

Kombinat
VEB ELEKTRO-APPARATE-WERKE
BERLIN-TREPTOW "FRIEDRICH EBERT"
Hoffmannstrasse 15-26
BERLIN
1193

Verantwortlicher Bearbeiter: Dr. L.Claßen

WAE/03-0001-02
Ausgabe: ODR II-15-14 B/241/89 3,0

Die vorliegende Dokumentation unterliegt nicht dem Änderungsdienst.

Spezielle Hinweise zum aktuellen Stand der P8000-Softwarepakete befinden sich in README-Dateien auf den Vertriebsdisketten.

Zum Programmier- und Entwicklungssystem P8000 existieren folgende Basisdokumentationsbände:

- Einführung in die Software des Gerätesystems P8000

- P8000-Hardwarehandbuch

- WEGA-Software Systemhandbuch (Administrator)
- WEGA-Software Programmierhandbuch (Reference)
- WEGA-Software Dienstprogramme (Utilities)

- UDOS-Software Systemhandbuch
- UDOS-Software Dienstprogramme
- UDOS-Software Mikroprozessorsoftware
- UDOS-Software Programmiersprachen

- OS/M-Software Systemhandbuch

Darüber hinausgehend wird für den im vorliegenden Dokumentationsband behandelten Problemkreis folgende weiterführende Literatur empfohlen:

- Claßen, Ludwig; Oefler, Ulrich
  Wissensspeicher Mikrorechnerprogrammierung
  VEB Verlag Technik Berlin, 1987

- Claßen, Ludwig; Oefler, Ulrich
  UNIX und C - Ein Anwenderhandbuch
  VEB Verlag Technik Berlin, 1987

- UDOS 1526 - Systemhandbuch
  VEB Buchungsmaschinenwerk Karl-Marx-Stadt, 1984

- Anleitung für den Bediener SCP 1520
  Anleitung für den Programmierer SCP 1520
  Anleitung für den Systemprogrammierer SCP 1520
  VEB Buchungsmaschinenwerk Karl-Marx-Stadt, 1985

Inhalt:

0. Einführung in das P8000-Konzept

Das aus einem 8-Bit- und einem 16-Bit-Mikrorechnerteil bestehende P8000 ist ein universell einsetzbares Programmier- und Entwicklungssystem für Multi-User-/Multi-Task-Anwendungen im ACS-Bereich (ACS Arbeitsplatzcomputersystem).
Die Leistungsfähigkeit jedes Mikrocomputers wird wesentlich durch sein Betriebssoftwaresystem bestimmt. Auf dem P8000 sind, um eine große Anzahl von Anwendungsgebieten zu erschließen, drei Betriebssysteme implementiert:

WEGA	(kompatibel UNIX	**)
UDOS	(kompatibel RIO	**)
OS/M	(kompatibel CP/M	**)

Auf dem 16-Bit-Mikrorechnerteil des P8000 ist das zum Betriebssystem UNIX System III kompatible Mehrbenutzer-Betriebssystem WEGA implementiert. Es stellt sowohl hinsichtlich seiner Leistungsfähigkeit als auch hinsichtlich seiner Anwendungsbreite eine neue Qualität gegenüber bisher bekannten Betriebssystemen für 8-Bit-Mikrorechner dar. Das Betriebssystem UNIX hat sich als ein internationaler Standard für 16-Bit- und 32-Bit-Mikrorechner durchgesetzt.
Auf dem 8-Bit-Mikrorechnerteil des P8000 sind, um die Aufwärtskompatibilität zu verfügbaren Softwaresystemen abzusichern, die Betriebssysteme UDOS und OS/M implementiert. Dadurch ist die Übernahme vorhandener, erprobter Softwarelösungen auf das P8000 möglich.
Das P8000 bietet durch seine Ausstattung dem Anwender ein breites Spektrum an Softwarearbeitsmöglichkeiten, das für vielfältige Problemlösungen eingesetzt werden kann.
Schwerpunkt der vorliegenden Realisierungsversion des P8000 Softwaresystems ist die Unterstützung der Softwareentwicklung für die Mikroprozessorfamilien:

U881/U882	Einchipmikrorechner
U880	8-Bit-Mikroprozessorsystem
U8001/U8002	16-Bit-Mikroprozessorsystem
K1810WM86	16-Bit-Mikroprozessorsystem.

Die Einbindung weiterer Mikroprozessorsysteme in das P8000-Entwicklungssystemkonzept ist möglich.

Für Echtzeitaufgaben in Anwendersystemen mit den 16-Bit-Mikroprozessoren U8001/ U8002 wird zusätzlich das hocheffektive Real-Time-Betriebssystem IRTS 8000 (Kernel, Monitor, Debugger, Handler ...) und das zugehörige - auf dem P8000 lauffähige - automatische Generierungssystem ICL 8000 bereitgestellt.

** Eingetragene Warenzeichen:

UNIX	Bell Laboratories
RIO	Zilog Co.
CP/M	Digital Research

1. Hardwarekonfiguration des Gerätesystems P8000

Des Gerätesystem P8000 besteht aus mehreren aufeinander abgestimmten Hardwareteilkomponenten, die, abhängig vom jeweiligen Einsatzfall, in verschiedener Weise miteinander konfiguriert werden können:

- P8000-Grundgerät mit 8- und 16-Bit-Mikrocomputerzentraleinheit, mit bis zu 1 MByte Hauptspeicher und mit zwei Floppy-Disk-Laufwerken (5 1/4 Zoll)

- P8000-Terminalarbeitsplatz mit alphanumerischem Zeichenvorrat und V.24- oder IFSS-Interface

- P8000-EPROM-Programmiermodul für EPROM-Schaltkreise der Typen 2708, 2716, 2732 und 2764 (EPROM electrically programmable read only memory).

- P8000-Hard-Disk-Beistellgerät (5 1/4 Zoll Winchesterlaufwerk)

- P8000-Matrixdrucker (EPSON LX86 oder ROBOTRON K63xx)

- P8000-In-Circuit-Emulatorsystem für die Mikroprozessorfamilien:

U881/U882	Einchipmikrorechner
U880	8-8it-Mikroprozessorsystem
U8001/U8002	16-Bit-Mikroprozessorsystem
K1810WM86	16-8it-Mikroprozessorsystem.

Neben diesen speziell für das Gerätesystem P8000 vorgesehenen Hardwarekomponenten können in P8000-Konfigurationen auch beliebige Terminals, Drucker, Rechnerkoppeleinheiten, Grafikarbeitsplätze u.a.m. verwendet werden, die den beim P8000 gegebenen Hardware- und Software-Interfacebedingungen genügen.

1.1. P8000-Grundgerät

Die Zentraleinheit des Programmier- und Entwicklungssystems P8000 ist in einem Kompaktgehäuse untergebracht. Eine Kartenbaugruppenaufnahme dient innerhalb das P8000-Grundgerätes zur mechanischen Fixierung von zwei durch Flachbandkabelstecker miteinander verbundenen Einzelleiterplatten mit dem 8- und 16-Bit-Mikrorechnerteil.
Auf der Leiterplatte des 16-Bit-Mikrorechnerteils befinden sich fünf 64-polige Steckverbinder, die zur Aufnahme von Speicherbaugruppen mit 64-K8it-DRAM-Speicherschaltkreisen dienen. Auf jeder dieser einzeln steckbaren Speicherbaugruppen ist ein 256-KByte-DRAM-Speicherbereich mit Paritätsfehlerüberwachung untergebracht (DRAM dynamischer RAM / RAM random eccess memory).
Die nicht mit Speicherbaugruppen belegten 64-poligen Steckverbinder auf dem 16-Bit-Mikrorechnerteil können zur Aufnahme zusätzlicher Ein-/Ausgabeerweiterungsbaugruppen genutzt werden.

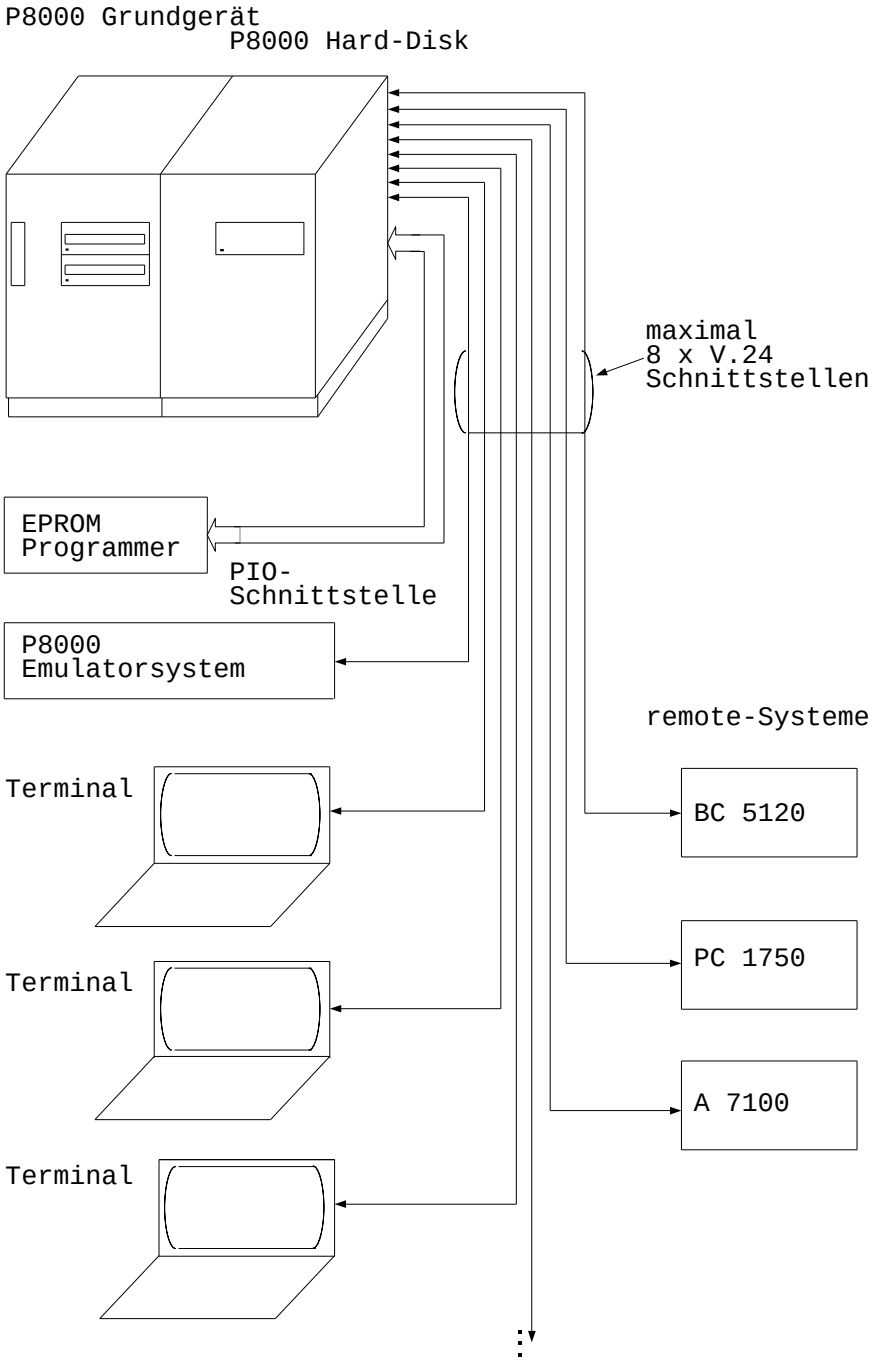

Bild 1.1. P8000-Gerätesystem

Neben den Elektronikbaugruppen befinden sich im P8000-Grundgerät zwei 5 1/4 Zoll Floppy-Disk-Lautwerke mit einer Speicherkapazität von jeweils bis zu 640 KByte und eine kompakte Stromversorgungseinheit.
Das P8000-Grundgerät ist standardmäßig mit acht seriellen und vier parallelen Interfaceschnittstellen ausgestattet, die zur Ankopplung von Terminalarbeitsplätzen, Hard-Disk Beistellgeräten, Druckern, In-Circuit-Emulatoren, EPROM-Programmiermodulen u.a.m. dienen können.
Im einzelnen existieren folgende Hardwaremerkmale des P8000-Grundgerätes:

- 8-Bit-Mikrorechnerteil auf Basis UA880 (4 MHz)
  - mit 2 Fassungen für EPROM-Schaltkreise der Typen 2716, 2732 oder 2764 (4, 8 oder 16 KByte)
  - mit 8 Stück U264 64-KBit-DRAM-Schaltkreisen (64 KByte)
  - mit 4 Stück U214 1Kx4-SRAM-Schaltkreisen (2 KByte)
  - mit 2 Stück UA856-SIO für vier serielle Schnittstellen (4x V.24 oder 4x IFSS - speziell für Terminal- und Druckeranschluß)
  - mit 1 Stück UA855-PIO für zwei 8-Bit-Parallelschnittstellen (speziell für EPROM-Programmiermodulanschluß)
  - mit 1 Stück UA858-DMA für direkten Speicherzugriff.
  - mit 1 Stück U8272-FDC für Floppy-Disk-Anschluß (5 1/4 Zoll FM- oder MFM-Aufzeichnungsverfahren und 8 Zoll FM- oder MFM-Aufzeichnungsverfahren)

- 16-Bit-Mikrorechnerteil auf Basis UB8001 (4 MHz)
  - mit 4 Fassungen für EPROM-Schaltkreise der Typen 2716, 2732 oder 2764 (8,16 oder 32 KByte)
  - mit 4 Stück U214 1Kx4-SRAM-Schaltkreisen (2 KByte)
  - mit 3 Stück UB8010-MMU für Speicherverwaltungsaufgaben
  - mit 2 Stück UA856-SIO für vier serielle Schnittstellen (4x V.24 oder 2x V.24 und 2x IFSS - speziell für Terminal- und Druckeranschluß)
  - mit 1 Stück UA855-PIO für zwei 8-Bit-Parallelschnittstellen (speziell für Hard-Disk-Anschluß)
  - mit 5 Steckverbindern für Speicher- und Ein-(Ausgabeerweiterungskarten (insgesamt bis zu 1 MByte)

- Speichererweiterungskarten mit je 38 Stück U264 64-KBit-DRAM-Schaltkreisen (256 KByte) mit Paritätsfehlerkontrolle

- 2 Stück in das P8000-Grundgerät integrierte 5 1/4 Zoll Floppy-Disk-Laufwerke mit jeweils bis zu 640-KByte-Speicherkapazität auf 80 Spuren (FM single density oder MFM double density) und Anschlußmöglichkeit für ein Floppy-Disk-Beistellgerät mit zwei Laufwerken 5 1/4 Zoll (FM und MFM) oder zwei Laufwerken 8 Zoll (FM und MFM).

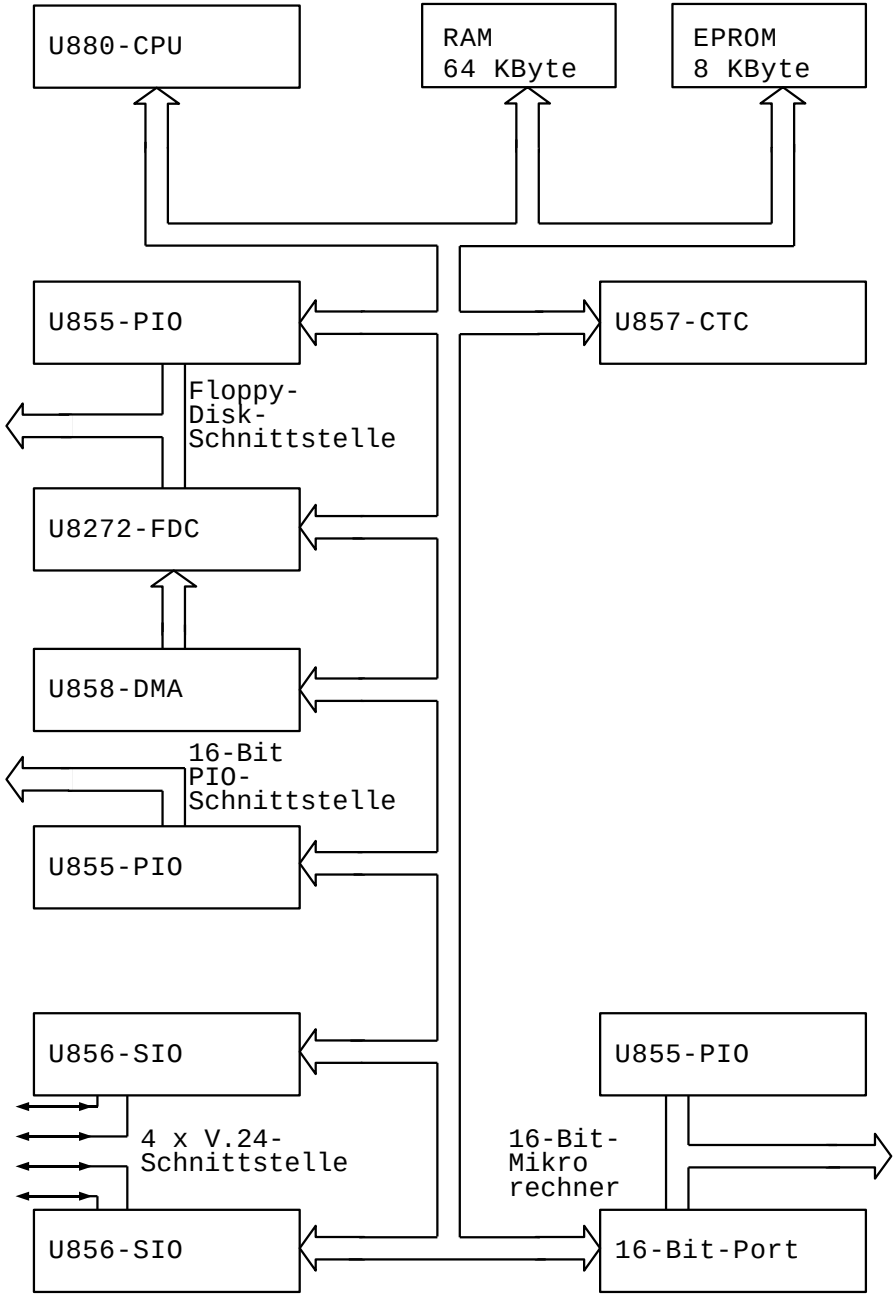

Bild 1.2. 8-Bit-Mikrorechnerteil P8000

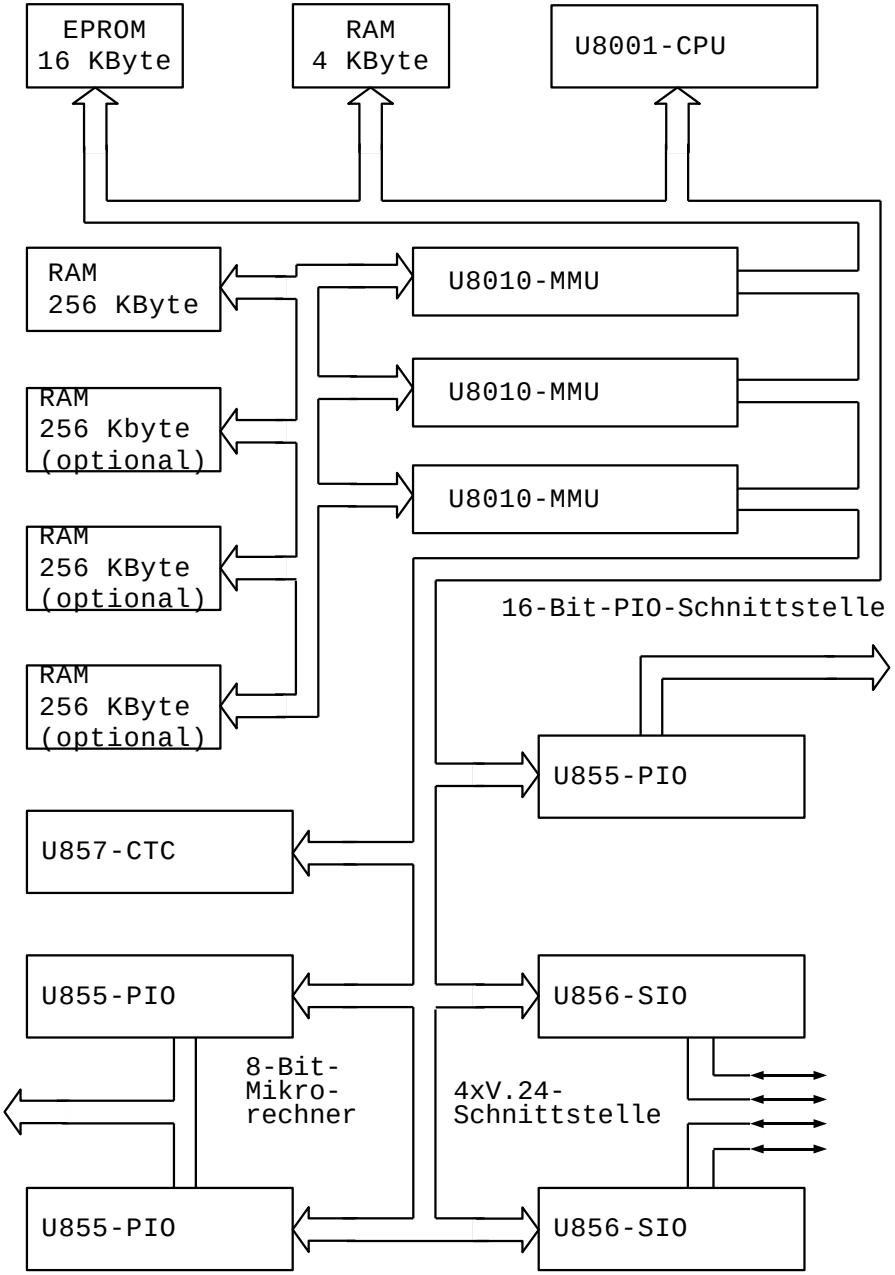

Bild 1.3. 16-Bit-Mikrorechnerteil P8000

Die Kommunikation zwischen dem 8- und 16-Bit-Mikrorechnerteil läuft über eine spezielle 32-Bit-Parallelschnittstelle mit begleitenden Handshake-Steuersignalleitungen.
Am P8000-Grundgerät befinden sich neben dem Netzschalter lediglich zwei Bedienelemente die Taste "RESET" Hardware-Reset) und die Taste "NMI" (NMI-Interrupt).
Zum P8000-Gerätesystem existiert ein ausführlicher Hardware-Dokumentationsband: "P8000 Hardwarehandbuch".

1.2. P8000-Terminalarbeitsplatz

Der P8000-Terminalarbeitsplatz besteht aus einer Eingabetastatur, einem Bildschirmmonitor und einem Terminalrechner der beides sowohl steuert als auch überwacht und nach außen eine V.24-/IFSS-Kommunikationsschnittstelle zum P8000-Grundgerät bereitstellt.
Der Zeichensatz des P8000-Terminalarbeitsplatzes entspricht standardmäßig dem ISO-7-Bit-Kode (ASCII-Kode).
Die Cursorsteuerzeichenfolgen des Bildschirmmonitors sind kompatibel mit dem Standardterminal ADM 31 und VT 100 - Standard: X 3.63 / ISO DP 6429.

1.3. P8000-EPROM-Programmiermodul

Zum Programmieren von EPROM-Schaltkreisen der Typen 2708, 2716, 2732 und 2764 ist im Gerätesystem P8000 ein EPROM-Programmiermodul vorhanden. Er wird über ein Kabel mit einem 25-poligen Steckverbinder an die PIO-Schnittstelle des 8-Bit-Mikrorechnerteils im P8000-Grundgerät angeschlossen, über die ihm Daten, Adressen und Steuerbefehle vorgegeben werden.
Einziges Bedienelement ist eine an der Oberseite befindliche Schwenkhebelfassung zur Aufnahme des zu programmierenden Speicherschaltkreises.

1.4. P8000-Hard-Disk-Beistellgerät

Im P8000-Hard-Disk-Beistellgerät sind ein oder zwei Winchester-Laufwerke (5 1/4 Zoll), die WDC-Anschlußsteuerung (WDC winchester disk controller) und ein Stromversorgungsmodul untergebracht.
Die Schnittstelle zwischen den Laufwerken und der Anschlußsteuerung entspricht dem Standard ST506/412.
Die Abmaße des Gehäuses des Beistellgeräts entsprechen denen des P8000-Grundgeräts.
Die Verbindung des P8000-Hard-Disk-Beistellgerätes mit dem P8000-Grundgerät erfolgt über die PIO-Schnittstelle des 16-Bit-Mikrorehnerteils mit einem 25-poligen Steckverbinder.

1.5. P8000-In-Circuit-Emulatorsystem

Die P8000-In-Circuit-Emulatorsysteme sind gesondert dokumentiert.

2. Installation des Gerätesystems P8000

die Installation einer P8000-Gerätekonfiguration kann unter Beachtung der gegebenen Hardwareinterfaceanschlüsse und unter Beachtung der jeweils vorliegenden Generierungsversion des Softwarebetriebssystems von jedem P8000-Nutzer weitestgehend selbständig erfolgen.
Die zentrale Rolle in jeder P8000-Gerätekonfiguration spielt dabei immer die Installation des P8000-Grundgerätes.
Ausführliche Hinweise zur Installation des Gerätesystems P8000 sind dem P8000-Hardwarehandbuch zu entnehmen.

2.1. Installation P8000-Grundgerät

Bei der Installation eines P8000-Gerätesystems sind, bezogen auf die Steckerbelegung an der Rückseite des P8000-Grundgerätes, die folgenden Festlegungen zu beachten:

Steckverbinderbuchse Geräteinterface

TTY0 (8-Bit-SIO 0 Kanal A) V.24/IFSS

WEGA: V.24-Terminalinterface (Multi-User)
UDOS: V.24-Rechnerkoppelinterface
OS/M: V.24-Rechnerkoppelinterface

TTY1 (8-Bit-SIO 0 Kanal B) V.24/IFSS

WEGA: V.24-Systemterminalinterface (*** Superuser ***)
UDOS: V.24-Systemterminal
OS/M: V.24-Systemterminal

TTY2 (8-Bit-SIO 1 Kanal A) V.24/IFSS

WEGA: V.24-Terminalinterface (Multi-User)
UDOS: Koppelinterface zu Einrichtungen der KEAW-electronic
OS/M: nicht benutzt

TTY3 (8-Bit-SIO 1 Kanal B) V.24/IFSS

WEGA: V.24-Terminalinterface (Multi-User) oder Druckeranschluß
UDOS: Druckeranschluß
OS/M: Druckeranschluß

TTY4 (16-Bit-SIO 0 Kanal A) V.24/IFSS

WEGA: V.24-Terminalinterface (Multi-User) oder Rechnerkoppelinterface oder P8000-Emulatorsystem

TTY5 (16-Bit-SIO 0 Kanal B) V.24/IFSS

WEGA: V.24-Terminalinterface (Multi-User)

TTY6 (16-Bit-SIO 1 Kanal A) V.24/IFSS

WEGA: V.24-Terminalinterface (Multi-User)

TTY7 (16-Bit-SIO 1 Kanal B) V.24/IFSS

WEGA: V.24-Terminalinterface (Multi-User) oder Rechnerkoppelinterface zu Einrichtungen der KEAW-electronic

PIO (8-Bit-PIO-1) EPROM

Parallelschnittstelle EPROM-Programmiermodul

PIO (16-bit-PIO 2) WINCHESTER

Parallelschnittstelle Hard-Disk-Beistellgerät

FDC (8-Bit-MR) FLOPPY

Anschluß externer Floppy-Disk-Beisteller 5 1/4 oder 8 Zoll

Eine andere Zuordnung der verfügbaren P8000-Geräteinterfaceanschlüsse ist prinzipiell möglich, bedarf jedoch der speziellen Softwaregenerierung

2.2. Installation P8000-Terminalarbeitsplatz

Die Installation des P8000-Terminalarbeitsplatzes erfolgt durch die Verbindung des Terminalrechners (Monitoruntersetzer) mit dem Monitor und der Tastatur über die dafür vorgesehenen Anschlüsse. Die Verbindung zwischen dem P8000-Terminalarbeitsplatz und dem P8000-Grundgerät wird über eine V.24-Schnittstelle mit einem 25-poligen Steckverbinder hergestellt.

2.3. Installation P8000-EPROM-Programmiermodul

Die Installation des P8000-EPROM-Programmiermoduls erfolgt durch die Verbindung mit dem 25-poligen Steckverbinder am P8000-Grundgerät (PIO-Schnittstelle 8-Bit-Mikrorechnerteil) über das zugehörige Anschlußkabel.

2.4. Installation P8000-Hard-Disk-Beistellgerät

Die Installation des P8000-Hard-Disk-Beistellgerät erfolgt durch Verbindung mit der Buchse WINCHESTER des P8000-Grundgeräts (Parallelschnittstelle des 16-Bit-Mikrorechnerteils) über das zugehörige Interfacekabel mit einem 25-poligen Steckverbinder.
Installationshinweise, die sich aus dem Typ des verwendeten Hard-Disk-Laufwerks ergeben, sind gesondert dokumentiert.

2.5. Installation P8000-In-Circuit-Emulatorsystem

Die Installation der P8000-In-Circuit-Emulatorsysteme ist gesondert dokumentiert.

3. Firmwarekomponenten des Gerätesystems

Die Softwareteile, die fest und unveränderlich in den EPROM-Speicherbauelementen des P8000-Gerätesystems abgelegt sind, werden als P8000-Firmware bezeichnet. EPROM-Speicherbauelemente mit Firmware befinden sich im P8000-Grundgerät, im P8000-Terminalarbeitsplatz, im P8000-Hard-Disk-Beistellgerät und im P8000-In-Circuit-Emulator.
Der P8000-EPROM-Programmiermodul arbeitet ohne eigene EPROM-Firmware.

3.1. Firmware P8000-Grundgerät

Die im P8000-Grundgerät befindlichen EPROM-Speicherbauelemente umfassen abhängig vom Bauelementetyp auf dem 8-Bit-Mikrorechnerteil einen Speicherbereich von 4, 8 oder 16 KByte und auf dem 16-Bit-Mikrorechnerteil einen Speicherbreich von 8, 16 oder 32 KByte.
Im einzelnen enthalten sie folgende Firmwarekomponenten:

8-Bit-Mikrorechner	Hardwareeigentest U880-Softwaremonitor 8-/16-Bit-Mikrorechnerkommunikation Anfangslader WEGA, UDOS und OS/M
16-Bit-Mikrorechner	Hardwareeigentest U8000-Softwaremonitor 8-/16-Bit-Mikrorechnerkommunikation Anfangslader WEGA

3.1.1. Hardwareeigentest

Das P8000-Grundgerät ist mit speziellen Programmen für den Hardwareeigentest ausgerüstet, die eventuelle Hardwarefehler durch Softwaremaßnahmen ermitteln.
Nach der allgemeinen mechanischen Inspektion der Elektronikbaugruppen, der Floppy-Disk-Laufwerke, der Interfacekabel usw. und nach der Verbindung des P8000-Grundgerätes mit dem 220V-Wechselstromnetz über eine Schutzkontaktleitung ist das P8000-Grundgerät prinzipiell einschaltbereit.
Zur Anzeige der Ausschriften des Hardwareeigentests und der verschiedenen Softwaresysteme muß allerdings zumindest noch ein betriebsbereites V.24- oder IFSS-Terminal (über die Buchse TTY1) an das P8000-Grundgerät angeschlossen werden.
Das Hardwareeigentestprogramm für den 8-Bit-Mikrorechnerteil wird automatisch nach dem Netzeinschaltreset oder manuell durch Eingabe des U880-Softwaremonitorkommandos "T" aktiviert (siehe Abschnitt 3.1.2.1.).

Achtung: Durch Betätigung der Taste 'RESET' am P8000-Grundgerät wird ebenfalls ein Rücksetzsignal erzeugt. In diesem Fall erfolgt jedoch - im Gegensatz zum Netzeinschaltreset - keine Aktivierung des Hardwareiegentestprogramms.

Das Hardwareeigentestprogramm des 8-Bit-Mikrorechnerteils im P8000 meldet sich mit der Ausschrift:

"P8000 Hardwaretest U880 - Version x.x"

Nach Abarbeitung des U880-Eigentests erfolgt grundsätzlich der Eintritt in den U880-Softwaremonitor.
Das Hardwareeigentestprogramm für den 16-Bit-Mikrorechnerteil wird vor dem automatischen Start des WEGA-Betriebssystems oder durch Eingabe des U8000-Softwaremonitorkommandos "T" aktiviert (siehe Abschnitt 3.1.2.2.).
Das Hardwareeigentestprogramm des 16-Bit-Mikrorechnerteils im P8000 meldet sich mit der Ausschrift:

"P8000 Hardwaretest U8001 - Version x.x"

Nach Abschluß des 16-Bit-Hardwareeigentests erfolgt die Ausschrift:

"SEGMENTED JUMPERS"
"MAXSEG= (xx)"

wobei 'xx' die Segmentnummer (hexadezimal) des maximal verfügbaren Speichersegmentes in der getesteten P8000-Hardwarekonfiguration darstellt.
Treten bei Hardwareeigentest des 16-Bit-Mikrorechnerteils Fehler auf, wird der Start des Betriebssystems WEGA verhindert und es erfolgt der Übergang in den U8000-Softwaremonitor.
Im einzelnen werden die folgenden Verifikationen übernommen:

8-Bit-Mikrorechnerteil	EPROM-CRC-Test
	SRAM -Speichertest
	UA855-PIO-Peripheriebausteintest
	UA857-CTC-Peripheriebausteintest
	UA856-SIO-Peripheriebausteintest
	U8272-FDC-Peripheriebausteintest
	UA858-DMA-Peripheriebausteintest
	DRAM -Speichertest
16-Bit-Mikrorechnerteil	EPROM-CRC-Test
	SRAM -Speichertest
	UA855-PIO-Peripheriebausteintest
	UA857-CTC-Peripheriebausteintest
	UA856-SIO-Peripheriebausteintest
	DRAM -Speichertest
	U8010-MMU-Test

Vor der Ausführung eines Testschrittes wird jeweils eine Testschrittnummer auf dem Terminal ausgegeben. Diese Testschrittnummer und die zugehörige Fehlernummer sind identisch. Treten Fehlfunktionen beim Hardwareeigentest auf, die die Fehlerausgabe verhindern, so bleibt zumindest die vor dem jeweiligen Test ausgegebene Testschrittnummer erhalten.

Werden Hardwarefehler vom Hardwareeigentestprogramm ermittelt, erfolgt auf dem angeschlossenen Terminal die Ausgabe einer Fehlermeldung der Form

"*** ERROR 'Fehlernummer' 'maximal 4 Fehlerparameter'"

In der Regel wird nach dem Auftreten eines Fehlers der Hardwareeigentest mit dem nächsten Testschritt fortgesetzt. Im einzelnen gilt die Fehlerliste im Anhang A für die Hardwareeigentestprogramme im P8000-Grundgerät.

3.1.2. U880- und U8000-Softwaremonitor

Die hardwarenahen Softwaretestfunktionen im EPROM-Softwaremonitor dienen zum Test von Programmkomponenten, deren Test so basisnah erfolgen muß, das jedes Betriebssystem den Test nur kompliziert, verfälscht oder gar unmöglich macht (z.B.: Interruptserviceroutinen, Gerätekanalprogramme usw.).

3.1.2.1. U880-Softwaremonitor

Der U880-Softwaremonitor gehört als einfaches Testhilfsmittel zum Betriebssystem UDOS. Er ist aber auch ohne UDOS voll arbeitsfähig.
Über die zum U880-Softwaremonitor gehörende Anfangsladeroutine (Lesen von Spur 0, Sektor 1 auf dem Laufwerk 0) sind alle 8-Bit-Betriebssysteme des P8000 vom U880-Softwaremonitor startbar.
Es existieren folgende Möglichkeiten zum Eintritt in den U880-Softwaremonitor:

- Eintritt in den U880-Softwaremonitor durch Netzeinschaltreset.
- Eintritt in den U880-Softwaremonitor durch Betätigung der Taste 'RESET'.
- Eintritt in den U880-Softwaremonitor aus dem Betriebssystem UDOS durch Eingabe des Kommandos 'DEBUG'.

Der U880-Softwaremonitor benötigt einen Programmspeicherbereich von 0 bis %0BFF und einen Arbeitsspeicher von %0C00 bis %0FFF (% Hexadezimalkennzeichen).
Dem Anwender steht der Speicher ab Adresse %1000 für seine zu testenden Programme zur Verfügung.
Nach dem Eintritt in den U880-Softwaremonitor erfolgt die Ausschrift:

"U880-Softwaremonitor Version x.x - Press RETURN "

Das Promptzeichen ('spitze Klammer zu') fordert zur Eingabe eines der U880-Softwaremonitorkommandos auf. Als Terminalkanal benutzt der U880-Softwaremonitor den SIO 0 / Kanal B des 8-Bit-Mikrorechnerteils (Buchse TTY1). Die Zeicheneingabe vom Terminal erfolgt interruptgesteuert, die Zeichenausgabe an das Terminal im Pollingbetrieb.
Kommandobeschreibung des U880-Softwaremonitors Anhang B.

3.1.2.2. U8000-Softwaremonitor

Zum Start des U8000-Softwaremonitors existieren die folgenden Möglichkeiten:

- Systemstart mit WEGA-Startdiskette
- Start des U880-Softwaremonitors und Aufruf des Kommandos "X"

Nur im ersten Fall steht für den U8000-Softwaremonitor das UDOS-Beriebssystem zur Verfügung, d.h. nur in diesem Fall können die Kommandos "GE", "S" und "O U" des U8000-Softwaremonitor ausgeführt werden.
Der U8000-Softwaremonitor benötigt einen (PROM-residenten) Programmspeicherbereich von 0 bis %3FFF und einen Arbeitsspeicher von %4000 bis %47FF (% Hexadezimalkennzeichen).
Dem Anwender steht der Speicher ab Adresse %8000 für seine zu testenden Programme zur Verfügung.
Nach dem Start des U8000-Softwaremonitors meldet er sich mit der Ausschrift:

"U8000-Softwaremonitor Version x.x - Press NMI"

Wird danach die NMI-Taste betätigt, so erfolgt ein automatischer Start des WEGA-Betriebssystems.
Zu beachten ist in diesem Fall, daß unter WEGA nur dann die Möglichkeit zur Arbeit mit Disketten besteht, wenn auf dem 8-Bit-Mikrorechnerteil des P8000 das Betriebssystem UDOS (auf der WEGA-Startdiskette) läuft. Der U8000-Softwaremonitor muß dann mit der WEGA-Startdiskette gestartet worden sein.
Das Betriebssystem WEGA kann vom U8000-Softwaremonitor aus außer über die Betätigung der NMI-Taste auch manuell mit dem Kommando "O" gestartet werden.
Soll das Betriebsystem WEGA nicht zur Abarbeitung gebracht werden, können die im Anhang C beschriebenen Kommandos des U8000-Softwaremonitors ausgeführt werden. Das Promptzeichen ('Stern' - "*") des U8000-Softwaremonitors fordert dabei zur Eingabe eines Kommandos auf.
Als Terminalkanal benutzt der U8000-Softwaremonitor den SIO 0 / Kanal B des 8-Bit-Mikrorechnerteils (Buchse TTY1). Der Datentransfer vom/zum Terminal erfolgt dabei über das 8-/16-Bit-Koppelinterface.
Sollte das Koppelinterface defekt sein ('Hardware Error in Connection') kann als Terminalkanal für den U8000-Softwaremonitor der SIO 0 / Kanal B des 16-Bit-Mikrorechnerteils benutzt werden (Buchse TTY5). In diesem Fall können die Kommandos "GE", "S", "Q", "QRES" und "O U" des U8000-Softwaremonitors nicht ausgeführt werden.
Die Terminaleingabe erfolgt interruptgesteuert, die Terminalausgabe im Pollingbetrieb. Die Übertragungsrate beträgt 9600 Baud.
Kommandobeschreibung des U8000-Softwaremonitors Anhang C.

3.1.3. P8000-Anfangsladeprozedur

Die automatische Anfangsladeprozedur der verschiedenen auf dem P8000-Grundgerät abarbeitbaren Betriebssysteme erfolgt nach einem einheitlichen Basisverfahren, das nach dem Start des U880-Softwaremonitors durch Drücken der Taste RETURN (nur nach RESET) bzw. durch Eingabe des Kommandos "O" aktiviert wird (siehe Abschnitt 3.1.2.1. und Anhang B).
Selektiert wird das einzulesende Betriebssystem einzig durch die eingelegte Systemladediskette im 5 1/4 Zoll Floppy-Disk-Laufwerk 0. Befindet sich hier eine UDOS-oder OS/M-Systemladediskette, wird eines dieser 8-Bit-Mikrorechnerbetriebssysteme eingelesen, geladen und gestartet. Befindet sich in diesem Laufwerk WEGA-Startdiskette mit der WEGA-Initialisierungsdatei (über OS.INIT von UDOS), wird der U8000-Softwaremonitor aktiviert und danach kann das 16-Bit-Mikrorechnerbetriebssystem WEGA geladen und gestartet werden.

Die Anfangsladeprozedur beinhaltet folgende Schritte:

1. Schritt Aktivierung der EPROM-Firmware zum Einlesen des 8-bit-Anfangsladers von der im Laufwerk 0 im P8000-Grundgerät befindlichen Systemladediskette des jeweiligen 8- oder 16-Bit-Beriebssystems (UDOS, OS/M oder WEGA).
2. Schritt Übergabe der Steuerung von der EPROM-Firmware an den eingelesenen 8-Bit-Anfangslader.
3. Schritt Abschalten des EPROM-Firmwarespeichers auf dem 8-Bit-Mikrorechnerteil des P8000-Grundgerätes.
4. Schritt Einlesen des 8-Bit-Betriebssystems UDOS oder OS/M durch den 8-Bit-Anfangslader.
5. Schritt Übergabe der Steuerung vom 8-Bit-Anfangslader an das eingelesene Betriebssystem UDOS oder OS/M.

Die Anfangsladeprozedur für die 8-Bit-Betriebssysteme ist hiermit beendet.
Die folgenden Schritte gelten nur für das Laden des 16-Bit-Betriebssystems WEGA:

6. Schritt Abarbeitung eines Initialisierungsprogramms unter Steuerung des 8-Bit-Betriebssystems UDOS zur Aktivierung des U8000-Softwaremonitors (EPROM-Firmware).
7. Schritt Nach Betätigung der NMI-Taste und Abarbeitung des Hardwareeigentests erfolgt das Einlesen des ersten 512-Byte-Anfangsladers (boot0) des Betriebssystems WEGA von der Hard-Disk.
8. Schritt Übergabe der Steuerung auf dem 16-Bit-Mikrorechnerteil des P8000-Grundgerätes von der EPROM-Firmware an den eingelesenen ersten WEGA-Anfangslader (boot0).
9. Schritt Abschalten des EPROM-Firmwarespeichers auf dem 16-Bit-Mikrorechnerteil des P8000-Grundgerätes.
10. Schritt Initialisierung eines zweistufigen Anfangsladevorganges zum Einlesen des WEGA-Betriebssystems

von der Hard-Disk durch den ersten 512-Byte-Anfangslader (boot0).
11.Schritt Zweistufiger Anfangsladevorgang des WEGA-Betriebssystems (boot und WEGA-Kernel).
12.Schritt Start des Betriebssystems WEGA

Außerdem kann der Anfangsladevorgang des Betriebssystems WEGA manuell über das Kommando "O" des U8000-Softwaremonitors eingeleitet werden (siehe Abschnitt 3.1.2.2. und Anhang C).

Die folgende Tabelle gibt eine Übersicht über die Möglichkeiten des Starts aller P8000-Betriebssysteme:

Start der 8-Bit-Betriebssysteme (UDOS, OS/M):

automatischer Start
- Starten des U880-Softwaremonitors
- Systemdiskette einlegen
- Drücken der Taste RETURN (nur nach RESET) bzw. Eingabe des Kommandos "O"
  (siehe 3.1.2.1. und Anhang B)

Start des 16-Bit-Betriebssystem (WEGA):

automatischer Start
- Starten des U8000-Softwaremonitors
- Drücken der Taste NMI
  (siehe 3.1.2.1. und Anhang C)

manueller Start
- Starten des U8000-Softwaremonitors
- Drücken der Kommandos "O"
  (siehe 3.1.2.2. und Anhang C)

3.2. Firmware P8000-Terminalarbeitsplatz

Die Firmware des P8000-Terminalarbeitsplatzes übernimmt die Steuerung des Monitors, der Tastatur und der seriellen Schnittstelle zum P8000-Grundgerät.

3.3. Firmware P8000-Hard-Disk-Beistellgerät

Das P8000-Hard-Disk-Beistellgerät enthält ein oder zwei 5 1/4 Zoll Hard-Disk-Laufwerke (Winchester) und die zugehörige WDC-Anschlußsteuerng.
Der Winchester-Disk-Controller (WDC) ist ein selbständiger Rechner. Er enthält den Mikroprozessor UA880 (4Mhz Taktfrequenz). Das Programm des WDC (Firmware) ist in 4-Kbyte-EPROM (2 x 2716) enthalten. Als Arbeits- und Datenspeicher sind 6-KByte-RAM (12 x U214) vorhanden. Zur Interrupterzeugung ist ein UA857-CTC vorgesehen.
Die Verbindung des WDC mit dem Laufwerk und dem P8000-Grundgerät (Host) erfolgt über entsprechende Schnittstellen.
Zwischen Laufwerk und WDC (Disk-Schnittstelle) wird eine

Schnittstelle ST506/412 realisiert.
Die Verbindung zwischen WDC und P8000-Grundgerät erfolgt über eine Parallelschnittstelle (Host-Schnittstelle). Hardwaremäßig ist die Host-Schnittstelle eine Parallelschnittstelle mit einem 8-Bit-Datenbus für beide Richtungen sowie zwei Handshakeleitungen zur Anpassung der Datenübertragung an einen UA855-PIO im P8000-Grundgerät. Außerdem existieren drei Leitungen zur Steuerung des Datenverkehrs und drei Statusleitungen zur Identifikation des Zustands der WDC-Anschlußsteuerung.
Softwaremäßig ist die P8000-Grundgeräteschnittstelle durch die Bedeutung der Statusbits und der möglichen Kommandos sowie durch die Datentransfers festgelegt. durch die Statusbits zeigt der WDC dem P8000-Grundgerät an, ob er bereit ist, über den 8-Bit-Datenbus Kommandos/Daten zu übernehmen bzw. Daten/Fehlernachrichten zu senden.

PIO-Interfacebelegung:

Port A:	D7	Datenbit 7
	.	
	.	
	.	
	D0	Datenbit 0
Port B:	D7	frei
	D6	Reset (für den WDC)
	D5	TE (Transfer Enable)
	D4	TR (Transfer Request)
	D3	frei
	D2	ST2 (Statusbit 2)
	D1	ST1 (Statusbit 1)
	D0	ST0 (Statusbit 0)

Im WDC sind die Serien-Parallel-Wandlung der Daten, die Synchronisationslogik für die seriellen Daten, die Markenerkennung/-einblendung sowie der CRC-Generator/-Checker enthalten.
Die P8000-Grundgeräte- und die Disk-Ansteuerung realisieren jeweils die nötigen Aktivitäten zwischen dem RAM und der zugehörigen Schnittstelle. Nach der Programmierung durch die CPU arbeiten beide Schnittstellen des WDC selbstständig und melden der CPU durch Interrupts die Beendigung der programmierten Aktivitäten.
Nach einem Hardwarereset des WDC (entweder vom Host oder nach dem Einschalten) wird ein Selbsttest durchgeführt. Die Schreib-Lese-Köpfe des Laufwerks werden auf den Zylinder 0 gefahren. die beiden Schnittstellen werden in einen Grundzustand versetzt und alle benötigte RAM-Zellen werden initialisiert. Anschließend programmiert die Firmware die P8000-Grundgeräteansteuerung so, daß die Schnittstelle auf der WDC-Seite bereit ist, 9 Bytes vom P8000-Grundgerät in den WDC-RAM zu übernehmen. Diese 9 Bytes werden vom WDC als Kommando mit zugehörigen Parametern interpretiert.
Die Firmware akzeptiert drei Arten von Kommandos: Lesen von der Hard-Disk, Schreiben auf die Hard-Disk, Formatieren einer Spur der Hard-Disk. Beim Schreiben wird anschließend

die Übergabe der zu schreibenden Daten an den WDC-RAM erwartet.
Nach der Positionierung der Schreib-Lese-Köpfe auf dem richtigen Zylinder wird die gewünschte Aktivität auf der Hard-Disk durchgeführt. Nach einer Leseoperation werden die gelesenen Daten an das P8000-Grundgerät übergeben. Anschließend geht der WDC wieder in Eingabebereitschaft oder er gibt an das P8000-Grundgerät einen aufgetretenen Fehler aus und geht dann in Eingabebereitschaft.
Durch das Formatieren einer Spur wird die betreffende Spur (ausgewählt durch Zylinder und Kopf) in geeigneter Weise für nachfolgende Aktivitäten (Lesen, Schreiben) vorbereitet. Die Daten werden MFM-codiert geschrieben. Der Spuranfang ist gekennzeichnet durch einen Indeximpuls. Anschließend werden 17 Sektoren pro Spur geschrieben.
Der Aufbau eines Sektors entspricht dem Standard EC 5057 mit 512 Byte Datenfeldlänge.
Stellt der WDC defekte Spuren fest, wird eine Ersatzspur automatisch aus einem dafür vorgesehenen Bereich zugewiesen.
Die formatierte Gesamtkapazität der angeschlossenen Laufwerke wird durch den WDC selbständig ermittelt und kann vom P8000-Grundgerät über ein entsprechendes Kommando abgefragt werden.

3.4. Firmware P8000-In-Circuit-Emulatorsystem

Die Firmwarekomponenten der P8000-In-Circuit-Emulatorsysteme wird in den zugehörigen Dokumentationsbänden gesondert dokumentiert.

4. Betriebssystemsoftware des Gerätesystems P8000

Die Betriebssystemsoftware des Gerätesystems P8000 wird ausführlich in den zugehörigen Softwaredokumentationsbänden beschrieben. Eine erste Übersicht über die Betriebssysteme des P8000-Grundgerätes dient hier zur Vermittlung der Softwarebasiskonzepte des P8000-Gerätesystems.

4.1. WEGA Betriebssystem

Das UNIX-kompatible Betriebssystem WEGA des P8000-Gerätesystems ist für die unterschiedlichsten Einsatzfälle konzipiert. Es ist ein Mehrbenutzer-Betriebssystem (Multi-User) mit Multitask-Eigenschaften, bei dem jeder Teilnehmer mehrere Prozesse (Programme) gleichzeitig bearbeiten lassen kann. Insgesamt sind bei der WEGa-Implementation auf dem P8000-Grundgerät bis zu acht quasisimultan an einer Zentraleinheit arbeitende Nutzer zulässig. Jeder Nutzer ist über einen Terminalarbeitsplatz mit dem P8000-Grundgerät verbunden.
WEGA beinhaltet ferner ein hierarchisches Dateiverwaltungssystem, Ein-/Ausgaberedirektion, Pipe- und Filterverarbeitungsmöglichkeiten, einen Shell-Kommandointerpreter, eine C-Sprachbasis u.a.m..
Auf Quellcodeebene ist WEGA zum UNIX System III und mit den zu UNIX kompatiblen Betriebssystemen PSU (ESER), MUTOS (CM4, K1600) und zur MUTOS-Implementation auf dem Bürocomputer A5120.16 kompatibel.
Die Nutzung des Betriebssystems WEGA setzt ein an das P8000-Grundgerät über ein PIO-Interface anschließbares Hard-Disk-Beistellgerät (5 1/4" Winchester-Hard-Disk) von minimal 10 MByte voraus. Die maximale Größe des externen Speichermediums ist nahezu unbegrenzt (mehrere 100 MByte). Als Backup-Medium für den externen Festplatten-Massendatenspeicher werden die im P8000-Grundgerät eingebauten 5 1/4 Zoll Floppy-Disk-Laufwerke verwendet.
Das P8000-Softwaresystem ist in der vorliegenden Realisierungsversion primär vorgesehen als Entwicklungssystem zur Programmentwicklung für die Mikroprozessorfamilien

U881/U882	Einchipmikrorechner
U880	8-Bit-Mikroprozessorsystem
U8001/U8002	16-Bit-Mikroprozessorsystem
K1810WM86	16-Bit-Mikroprozessorsystem

und als Zentralgerät zum Anschluß der entsprechenden P8000-In-Circuit-Emulatorversion zum Software-Integrationstest.

Das P8000 ist darüber hinaus aber auch sofort ohne Zusatz oder Änderung einsetzbar für

- Aufgaben bei der Rationalisierung und Automatisierung der Büro- und Verwaltungsarbeit
- Basiskonfigurationen beim Aufbau von komplexen, allen Anforderungen entsprechenden Datenbanksystemen
- Textverarbeitungsaufgaben bis hin zur Aufbereitung von Texten für den Lichtsatz

- Zentralrechnersysteme, die dezentrale Echtzeitsteuer- und -regelsysteme überwachen und Dateiarbeit ausführen können
- Rechnerkopplungen des P8000-Grundgerätes mit gleichartigen oder mit unterschiedlichen Rechnern, wie Bürocomputern und Kleinrechnern sowie für Kopplungen des P8000-Grundgerätes über ein Modem an ein Datennetz
- Unterstützungsaufgaben bei der Entwicklung von Compilern für spezielle Sprachen (Fachsprachen der Mess-, Steuer- und Regelungstechnik, spezielle Prüfautomatensprachen ...) durch Compiler-Entwicklungssoftware.

Das Betriebssystem WEGA kann in weiteren Softwarearbeitsetappen aber auch eingesetzt werden für

- CAD/CAM-Entwurfsarbeitsplätze durch zusätzliche Bereitstellung von grafikfähigen Verarbeitungsprogrammen
- menügesteuerte Spezialarbeitsplätze unter WEGA-Steuerung die keine besonderen Qualifikationsmerkmale von dem Bedienpersonal fordern

und für vieles andere mehr.

Achtung: Die Nutzung des Betriebssystems WEGA setzt unbedingt einen Software-Systemverwalter (Superuser) voraus, der den Aufbau, die Struktur und die Arbeitsweise sowie die Pflege und Wartung dieses Betriebssystems beherrscht. Anderenfalls ist eine effektive Arbeit mit diesem hochkomplexen Softwaresystem nicht möglich. Die Betriebsweise eines Programmier- und Entwicklungssystems P8000 unter Steuerung des Betriebssystems WEGA unterscheidet sich grundsätzlich von der Betriebsweise eines 8-Bit-Mikrorechners mit einem Floppy-Disk-Betriebssystem.

4.1.1. WEGA Struktur des Betriebssystems

Das WEGA-Softwaresystem des P8000-Gerätesystems besteht aus folgenden Komponenten:

- Anfangsladeprogramm und Standalone-Programme, die unter Steuerung des Anfangsladers arbeiten
- WEGA-Betriebssystemkern
- System-, Dienst- und Anwenderprogramme, die unter Steuerung des Betriebssystems WEGA arbeiten.

Nach dem Netzeinschalten und nach jedem Hardware-RESET ist die Software im EPROM-Firmwarespeicher des P8000-Grundgerätes aktiv. In ihr sind Debug-Funktionen des U880-/U8000-Softwaremonitors für die maschinennahe Softwarefehlersuche, ein Hardwareeigentestprogramm, Programme zur Abwicklung der Kommunikation zwischen 8- und 16-Bit-Mikrorechner im P8000-Grundgerät sowie ein Programm zur automatischen Initialisierung des WEGA-Anfangsladers enthalten.
Der Ladeprozeß des Betriebssystems WEGA erfolgt grundsätzlich von der Festplatte im Hard-Disk-Beistellgerät.

Über die im P8000-Grundgerät eingebauten Floppy-Disk-Laufwerke kann die Festplatte bei Bedarf mit Hilfe der Standalone-Programme neu eingerichtet werden.

Anfangsladeprogramm - Der Anfangsladevorgang des Betriebssystems WEGA besteht aus mehreren Schritten. Von der EPROM-Firmware des 16-Bit-Mikrorechners im P8000-Grundgerät wird ein erster Anfangslader (boot0 512 Byte lang) von der Festplatte eingelesen und zur Abarbeitung gebracht. Von diesem Programm wird selbständig das eigentliche WEGA-Anfangsladeprogramm (boot) eingelesen und gestartet.
Unter Steuerung des WEGA-Anfangsladerprogrammes (boot) wird danach der WEGA-Betriebssystemkern eingelesen. Er übernimmt in der Folge alle weiteren Systeminitialisierungen.
Der gesamte Ladevorgang kann manuell oder automatisch ausgeführt werden. Im ersten Fall erfolgen alle Schritte unter Steuerung des Bedieners. Das Betriebssystem WEGA geht beim manuellen Systemstart zunächst in den Single-User-Mode. Beim automatischen Systemstart wird der Ladevorgang nur initiiert, das Betriebssystem WEGA geht hier sofort in den Multi-User-Mode über.

WEGA-Betriebssystemkern - Das Betriebssystem WEGA kennt - wie schon angedeutet - zwei Betriebsarten, den Single-User-Mode und den Multi-User-Mode.
Im Single-User-Mode ist nur die V.24-Systemkonsole des WEGA-Systemprogrammierers (WEGA-Superuser) aktiv. In dieser Betriebsart wird die Softwarewartung des Systems durchgeführt. Diese Betriebsart kann auch gewählt werden, wenn das P8000 als Einbenutzer-System eingesetzt werden soll.
Im Multi-User-Mode werden alle angeschlossenen und eingeschalteten Terminals aktiviert. Bis zu acht Nutzer können über separate Terminalarbeitsplätze in dieser Betriebsart quasisimultan an einem P8000-Grundgerät unter WEGA arbeiten.

Folgende Ein-/Ausgabegeräte sind standardmäßig unter Steuerung des WEGA-Betriebssystemkerns verfügbar:

- 8 serielle V.24-Kanäle für die WEGA-Systemkonsole, für Anwenderterminals, für einen Drucker (Type: Epson LX86 oder robotron K63xx mit V.24-Schnittstelle) und zur Kopplung des P8000 mit weiteren Rechnern
- 1 oder 2 Festplattenlaufwerke (P8000-Hard-Disk-Beistellgerät) mit jeweils maximal bis zu 10 logischen Dateisystemen angeschlossen über einen Hard-Disk-Controller mit Parallelinterface
- 2 Floppy-Disk-Laufwerke 5 1/4 Zoll (MFM-Aufzeichnungsverfahren, 80 Spuren mit 32 Sektoren mit 256 Byte) als Back-Up-Medium.

Neben diesen Standardgeräten kann der Anwender seinen speziellen Erfordernissen entsprechend zusätzliche Ein-/Ausgabegeräte in das Betriebssystem einbinden.

4.1.2. WEGA Standalone-, System- und Dienstprogramme

die Standalone-, System- und Dienstprogramme des Betriebssystems WEGA bilden die Softwarebasis des P8000-Anwenders. Sie unterstützen ihn bei der Lösung seiner speziellen Aufgabenstellung in einem breiten Bereich.

WEGA Standalone-Programme

Die Standalone-Programme dienen zur Initialisierung des WEGA-Softwaresystems. Ihre Nutzung ist nur beim Neustart, bei Systemzusammenbrüchen und bei Softwarewartungsarbeiten notwendig. Die Standalone-Programme arbeiten unter Steuerung des WEGA-Anfangsladeprogrammes (boot):

boot	Anfangslader (boot)
format	Formatierungsprogramm Hard-Disk
verify	Überprüfngsprogramm Hard-Disk
mkfs	Einrichten eines WEGA-Dateisystems
install	Laden des Betriebssystems WEGA (WEGA Back-up Floppy-Disks im UDOS-Format)
cat	Ausgabe des Hard-Disk-Inhaltes
shipdisk	Vorbereitung der Hard-Disk zum Transport
diags	erweitertes Hardware-Eigentestprogramm

WEGA Systemprogramme

Die WEGA-Systemprogramme bilden ein Grundbausteinsortiment, das der Anwender standardmäßig bei der Lieferung des Betriebssystems WEGA komplett übergeben bekommt (WEGA-Kommandosatz).
Die insgesamt ca. 280 WEGA-Systemprogramme überstreichen breite Anwendungsbereiche von der Rechnerkommunikation über die Textverarbeitung bis zur Softwareentwicklungsunterstützung.
Die WEGA-Systemprogramme sichern außerdem den Zugriff des Anwenders zu den gesamten WEGA-Betriebssystemressourcen.
Zum WEGA-Standardsoftwarekommandosatz gehören insbesondere folgende Systemprogramme (Auswahl):

Systemzugriffskontrolle

login		Nutzerzuschaltung
newgrp		Ändern der Gruppenzugehörigkeit
passwd		Schlüsselwortvergabe
su		Erweitern der Benutzerzugriffsrechte
sync	S	Aktualisieren des Superblocks

Systemverwaltung/Statusinformation

date		Ausgabe von Datum und Uhrzeit
dd		Konverieren/Umkopieren
du		Ermittlung der Plattenspeicherbelegung
fsck	S	Dateisystemkonsistenzüberprüfung
man		Ausgabe einer Kommandobeschreibung
mkfs	S	Anlegen eines Dateisystems
mknod	S	Anlegen einer Ein-/Ausgabedatei
mount	S	Mount/Dismount eines Dateisystems
ps		Prozeßstatusermittlung
pstat	S	Ausgabe von Systemtabellen
quot	S	Ausgabe der Dateisystemnutzerblöcke

Terminal- und Druckerarbeit

echo	Echoausgabe
lpr	Druckerausgabeprogramm
stty	Setzen von Terminalparametern
tabs	Setzen Terminaltabulatoreinstellung
tty	Ermittlung des Terminalnamens
who	Wer arbeitet am System?

Kommandointerpreter

csh	C-Shell Kommandointerpreter
sh	Shell-Kommandointerpreter

Dateiverwaltung

cat		Dateiausgabe
cd		Wechsel des Arbeitsdirectories
chmod		Setzen des Dateizugriffserlaubnisbits
chown		Wechsel der Besitzerzugehörigkeit
chgrp		oder der Gruppenzugehörigkeit
clri	S	Löschen einer i-node
cmp		Vergleich zweier Dateien
comm		Suchen gleicher Zeilen in Dateien
cp		Kopieren von Dateien
crypt		Verschlüsseln/Entschlüsseln
dcheck	S	Konsistenzprüfung von Directories
df		Ausgabe freier Dateisystemblöcke
diff		Ermittlung von Dateiunterschieden
file		Ermittlung eines Dateityps
find		Suchen von Dateien
ln		Herstellen einer Dateilinkverbindung

ls		Ausgabe des Directoryinhaltes
mkdir		Anlegen eines Directories
mv		Verschieben / Umbenennen von Dateien
od		Dumpausgabe
pwd		Ausgabe des Arbeitsdirectories
rm rmdir		Löschen von Dateien und Directories
split		Aufsplitten einer Datei
sum		Bestimmung Blockanzahl/Prüfsumme
tail		Übergehen von Dateiteilen
touch		Aktualisierung Dateidatum/-zeit

Kommunikation

LOAD		Programm vom P8000 in den Emulator laden
SEND		Programm vom Emulator in das P8000 übertragen
getfile		Einlesen von Dateien von einem Remote-System
local		Rückkehr zum lokalen Remote-System
putfile		Übertragung von Dateien z einem Remote-System
remote		Zuschaltung übergeordnetes WEGA-Systems
cu		WEGA-Rechnerkopplung
enroll		Schlüsselwortvergabe.
xsend xget		Senden und Empfangen geheimer Post
mail		Senden und Empfangen von Post
mesg		Nachrichtenempfangssperre
uucp		WEGA-WEGA Dateikopierprogramm
uux		WEGA-WEGA Kommandoausführung
wall	S	Schreiben an alle Nutzer
write		Schreiben an andere Nutzer

Softwareprojektunterstützung

ar	Archiv- und Bibliotheksverwaltung
cal	Kalenderausgabe
calendar	automatischer Terminkalender
make	Programmerzeugungshilfe
tar	Magnetbandarchivar

Programmabarbeitungsunterstützung

at	zeitgesteuerte Kommandoabarbeitung
expr	Berechnung arithmetischer Ausdrücke
kill	Prozeßabarbeitungsabbruch
nice	Wechsel der Abarbeitungspriorität
nohup	
prof	Ausgabe Programmabarbeitungsprofil
sleep	Abarbeitungspause
tee	Ein-/Ausgabepipevermittlung

time	Bestimmung einer Ausführungszeit
true	Erzwungener Exitstatus TRUE/FALSE
units	Einheitenumrechnungsprogramm
wait	Warten auf einen Hintergrundprozeß

Programmiersprachen

adb	interaktiver U8000-Programmtestdebugger
as	U8000-PLZ/ASM-Assemblerprogrammübersetzer
bc	Arithmetik-Tischrechnerprogramm
cas	U8000-Standard-Assemblerprogrammübersetzer
cb	C-Programm Formatierungshilfe
cc	U8000-C-Compiler (nichtsegmentiert)
dc	Tischrechnerprogramm
ld	Lader (nichtsegmentiert)
lex	Programmgenerator für lexik. Analyse
lint	C-Syntaxprüfer
lorder	Ermittlung von Objektprogrammbezügen
m4	Makroprozessor
nm	Symboltabellenausgabe
plz	PLZ/SYS-Compilerdriver
plzcg	PLZ/SYS-Kodegenerator
plzsys	PLZ/SYS-Compiler
scc	U8000-C-Compiler (segmentiert)
sld	Lader (segmentiert)
size	Ausgabe einer Objektprogrammgroße
strip	Entfernen der Symboltabelle
yacc	Compiler-Compiler

Textverarbeitung

awk	Textfilterprogramm
col	Spaltendrucktextfilter
deroff	Entfernen aller Formatierungsbefehle
ed	Texteditor
eqn	Ausgabe mathematischer Formeln
ex	Texteditor
grep	Durchmustern von Dateien
egrep	erweitertes Durchmustern
fgrep	beschleunigtes Durchmustern
join	identifikationszeichenbedingte Ausgabe
look	Suchen von Dateizeilen
pr	formatierte Dateiausgabe
ptx	Wortindexerstellung
sed	Datenstromeditor
sort	Sortierprogramm
spell	Suchen von Rechtschreibfehlern
tbl	Tabellenformatierungsprogramm
tr	Zeichenkettenkonvertierung
troff	Textformatierung
nroff	
uniq	Löschen doppelter Dateizeilen

vi	bildschirmorientierter Texteditor
wc	Wortzählungsprogramm

Anmerkung: Die mit einem 'S' gekennzeichneten Systemprogramme sind sogenannte Superuser-Kommandoprogramme, über die nur der WEGA-Systemprogrammierer verfügen kann.

Weitere Systemprogramme, wie ein nutzerbezogenes Rechenzeitabrechnungsprogramm, ein Fortschreibungssystem für Anwenderprogramme (sccs source code control system), ein Bildschirmmenügenerierungssystem u.v.a.m. kann der Anwender in Abhängigkeit von der Belegung seines Hard-Disk-Plattenspeichers nutzen.

WEGA Dienstprogramme

Zur Abarbeitung unter Steuerung des WEGA-Softwaresystems werden folgende Dienstprogrammpakete für die Mikroprozessorsoftwareentwicklung, für Datenbankverwaltungssysteme, für die grafische Datenverarbeitung, für CAD/CAM-Aufgabenstellungen, für die Rechnerkommunikation und für vieles andere mehr bereitgestellt:

WEGA-CROSS	Cross-Software U880, U881/2/4, K1810WM86	
WEGA-PASCAL	PASCAL-Compiler	
WEGA-FORTRAN77	FORTRAN77-Compiler	
WEGA-BASIC	BASIC-Compiler und -Interpreter	
WEGA-COBOL	COBOL-Compiler	
WEGA-REMOTE	Fernverarbeitung SCP, DCP, UDOS, WEGA(UNIX)	
WEGA-EMSCP	Multi-User SCP-Emulator	
WEGA-DATA	Datenverwaltungssystem	
WEGA-CALC	Tabellenkalkulationsprogramm	
WEGA-WORD	Textverarbeitungsprogramm	
WEGA-IGE	Grafikeditorsystem	
WEGA-3D/GKS	3D-Grafik-Kernel-System	
IRTS/ICL-8000	IRTS-8000 I	Echtzeitbetriebssystem U8001/U8002 (UDOS-Basis)
IRTS/ICL-8000	IRTS-8000 II	Echtzeitbetriebssystem U8001/U8002 (WEGA-Basis)
IRTS-880	IRTS-880	Echtzeitbetriebssystem U880 (EPROM-Basis)
IRTS-1810WM86	IRTS-1810WM86	Echtzeitbetriebssystem K1810WM86 (EPROM-Basis)

u.a.m..

Kurzinformation zu ausgewählten WEGA Dienstprogramme:

WEGA-CROSS - Cross-Mikroprozessorsoftwareentwicklungspakete für die Ein-Chip-Mikrorechnerfamilie U881/U882, für die 8-Bit-Mikroprozessorfamilie U880 und für die 16-Bit-Mikroprozessorfamilie K1810WM86.

Bestehend aus: U881/U882-PLZ/ASM-Cross-Assembler, LOAD-/SEND-Prozeduren für Emulator oder Entwicklungsmodul, Installationsunterstützungssoftware, On-Line-Dokumentation, On-Line-Manual, EPROM-Programmierunterstützungssoftware.

U880-PLZ/ASM-Cross-Assembler, LOAD/SEND-Prozeduren für Emulator oder Entwicklungsmodul, Installationsunterstützungssoftware wahlweise UDOS-Objektprogrammgenerierung, On-Line-Dokumentation, On-Line-Manual, EPROM-Programmierunterstützungssoftware.
U880-C-Cross-Compiler mit Optimierer und Standardbibliothek, U880-PLZ/ASM-Quellkodegenerierung.

U880-Turbo-Cross-Assembler mit bedingter Assemblierung, genesteten/rekursiven Makroaufrufen, External-/Globalvariablen, Operandenarithmetik und Mnemonikredefinition.
Lademodulgenerierung für verschiedene Formate (a.out, I-HEX).
LOAD/SEND-Prozeduren für Emulator oder Entwicklungsmodul, On-Line-Manual, Installationsunterstützungssoftware, EPROM-Programmierunterstützungssoftware.

K1810WM86-Turbo-Cross-Assembler mit bedingter Assemlierungg, genesteten/rekursiven Makroaufrufen, External-/Globalvariablen, Operandenarithmetik und Mnemonikredefinition.
Lademodulgenerierung für verschiedene Formate (a.out, I-HEX, MS-DOS und CP/M-86).
Intelligenter U880-Quellkodetranslator für K1810WM86.
LOAD/SEND-Prozeduren für Emulator oder Entwicklungsmodul, On-Line-Manual, Installationsunterstützungssoftware, EPROM-Programmierunterstützungssoftware.

Anmerkung:
Das Mikroprozessorsoftwareentwicklungspaket für die 16-Bit-Mikroprozessorfamilie U8001/U8002 (C-Compiler, Assembler, Lader und PLZ/SYS-Compiler, PLZ/ASM-Assembler, Lader) ist im Basislieferumfang des Betriebssystems WEGA standardmäßig enthalten.

Es besteht aus:

cas	U8000-Assembler	(WEGA-Standardassembler für segmentierte und nichtsegmentierte Programme)
cc	U8000-C-Compiler	(C-Compiler für nichtsegmentierte Programme)
scc	U8000-C-Compiler	(C-Compiler für segmentierte Programme)
ld	Lader	(Lader für nichtsegmentierte Programme)
sld	Lader	(Lader für segmentierte Programme)

as	U8000 PLZ/ASM-Assembler
plz	U8000 PLZ/SYS-Compiledriver
plzsys	U8000 PLZ/SYS-Compiler
plzcg	U8000 PLZ/SYS-Kodegenerator

Dabei existieren folgende Kombinationsmöglichkeiten:

cas	arbeitet mit	ld und sld	zusammen
cc	arbeitet mit	cas und ld	zusammen
scc	arbeitet mit	cas und sld	zusammen
as	arbeitet mit	ld	zusammen
plz	arbeitet mit	plzsys und plzcg	zusammen
plzsys	arbeitet mit	plzc	zusammen

WEGA-IGE - Basissoftwarepaket für grafische Datenverarbeitungsaufgaben.

Bestehend aus:	ige	- Interaktiver Farbgrafikeditor mit menügesteuerter Benutzerschnittstelle.
	gcat.c	- Ausgabeprogramm für farbgrafische Bilder.
	setcolor.c	- Farbsteuerung von Schriftzeichen.
	grdef.h	- INCLUDE-Datei für die programmgesteuerte Erzeugung von farbgrafischen Bildern in C-Programmen.
	grlib.c	- Standardbibliotheksroutinen für die farbgrafische Programmierung von Bildern.

Für alle Module gilt: ANSI-Steuerzeichenbasis X 3.63 / ISO DP 6429 - kompatibel 41xx-Serie.
Die Auslieferung des interaktiven graphischen Editors 'ige' erfolgt in Quell- und Maschinenkode, so daß der Anwender Modifikationen oder Erweiterungen seinen Erfordernissen entsprechend selbst vornehmen kann.

4.2. UDOS Betriebssystem

Das Betriebssystem UDOS ist ein Floppy-Disk-Betriebssystem für den 8-bit-Mikrorechnerteil des P8000. Es zeichnet sich durch ein komfortables Dateiverwaltungssystem, durch die wahlfreie Zuordnung von Ein-/Ausgabedatenströmen, durch die automatische Speicherplatzverwaltung und durch einen umfangreichen Kommandosatz aus.
Unter Steuerung des Betriebssystems UDOS ist insbesondere Entwicklungssoftware für die in der DDR produzierten Mikroprozessorfamilien U880, U881/U882/U883 und U8000 verfügbar. Daneben existieren eine Fülle weiterer UDOS-Softwarepakete von Compilern für höhere Programmiersprachen über Makroprozessoren bis hin zur Textverarbeitung.
Das auf dem 8-Bit-Mikrorechnerteil des P8000 laufende UDOS ist mit dem für den Bürocomputer A5120 und dem Personalcomputer PC1715 existierenden UDOS im grundsätzlichen Aufbau identisch. Im Diskettenformat unterscheiden sich nur die A5120-Implementation, da auf dem P8000 und dem PC1715 ein Sektorformat gemäß dem Standardformat 34 (MFM / 32 Sektoren mit 256 Byte) realisiert wird.
Vom Betriebssystem UDOS werden standardmäßig folgende Ein-/Ausgabegeräte des P8000 bedient:

- ein serieller V.24-Kanal für die UDOS-Systemkonsole
- ein bis maximal vier Floppy-Disk-Laufwerke 5 1/4 Zoll
- ein Drucker (Type: Epson LX86 oder robotron K6312 mit V.24-Schnittstelle)
- EPROM Programmiereinrichtung

Weitere Ein-/Ausgabegeräte können vom Anwender seinen speziellen Erfordernissen entsprechend problemlos in das UDOS-Softwarekonzept eingebunden werden.
Das Betriebssystem UDOS verwaltet einen Operationsspeicherbereich (RAM) von 64-KByte.

4.2.1. UDOS Struktur des Betriebssystems

Das Betriebssystem UDOS besteht aus dem U880-Softwaremonitor SMON, dem 6-KByte Operationssystemkern OS, dem 8-KByte Dateiverwaltungssystem NDOS und 27 externen UDOS-Standardkommandos.
Zu diesen Grundkomponenten kommt eine Fülle weiterer Softwarepakete für ausgewählte Anwendungslinien.
Dabei ist zu beachten, daß der Nutzer, seinen speziellen Erfordernissen und Wünschen entsprechend, jederzeit neue Kommandos kreieren und problemlos in das UDOS-Konzept einfügen kann.

EPROM-Firmware - Im EPROM-Firmwarespeicher sind Hardwareeigentestprogramme und der U880-Softwaremonitor enthalten. Zum U880-Softwaremonitor gehören die Hardware-/Softwareinitialisierung, UDOS-Anfangsladerstart, Konsoltreiber, Debug-Kommandos und ein Floppy-Disk-Treiberprogramm.
Für den Anfangsladerstart wird vom U880-Softwaremonitor ein Parametervektor zum Einlesen der eigentlichen UDOS-Urladerroutine von der Systemdiskette aufgebaut. Beim Systemstart

wird der Urlader dann von der UDOS-Systemdiskette eingelesen und abgearbeitet.
Das Eigentestprogramm realisiert nach jedem Spannungseinschalten einen EPROM-Test, RAM-Test, Interrupttest und den Test der Peripherieschnittstellen. Bildschirmausschriften geben dem Nutzer Auskunft über die Betriebsbereitschaft des Gerätes oder über eventuell vorhandene Hardware-Fehler.
Der Konsoltreiber steuert den Datenverkehr zwischen UDOS und dem Bedienerkommunikationsgerät über ein V.24-Interface. Das Kernstück des V.24-Konsoltreibers bilden die Programmteile PUTA und GETA, die die Einzelzeicheneins-/Ausgabe realisieren.
Im U880-Softwaremonitor sind außerdem Software-Debug-Kommandos für den maschinennahen Softwaretest, wie das Lesen und Schreiben von Speicher- und Registerwerten, das Transferieren von Speicherbereichen, das Setzen eines Software-Haltepunktes und den Programmstart enthalten. Für diese Testfunktionen existiert ein eigener Kommandointerpreter.
Der Floppy-Disk-Treiber des Betriebssystems UDOS ermöglicht die Arbeit mit 5 1/4-Zoll-FD-Laufwerken doppelter Schreibdichte (MFM, 40 oder 80 Spuren).
Die Behandlung von 40-Spur-Disketten auf den 80-Spur-Laufwerken erfolgt durch Umschaltung der angeschlossenen 80-Spur-Laufwerke.
Das Betriebssystem UDOS arbeitet grundsätzlich mit softsektorierten Minidisketten (5 1/4"). Durch den Einsatz des Floppy-Disk-Controller-Schaltkreises U8272 Im P8000 können jedoch - falls notwendig - auch alle Diskettenformate entsprechend den Standardformaten 2740, 34 und KROS 5110/01 durch spezielle Kanalprogramme realisiert werden.
Für das Betriebssystem UDOS des P8000 gelten im einzelnen folgende Diskettenformate:

5 1/4" SS DD: Spur 0-39; 16 Sektoren a' 256 Byte; 160 KByte
5 1/4" SS DD: Spur 0-79; 16 Sektoren a' 256 Byte; 320 KByte
5 1/4" DS DD: Spur 0-79; 32 Sektoren a' 256 Byte; 640 KByte

Maximal sind vier Floppy-Disk-Laufwerke, gekennzeichnet mit den logischen UDOS-Gerätenummern 0-3 anschließbar. Der Systemstart erfolgt grundsätzlich vom Laufwerk 0.
Die UDOS-Systemkonsole wird über eine V.24-Schnittstelle des 8-Bit-Mikrorechnerteils im P8000 angeschlossen.

Operationssystemkern OS - Das Operationssystem OS des Betriebssystems UDOS führt die Speicherplatzverwaltung während aller Abarbeitungsphasen von Dienst- und Anwenderprogrammen durch. Es realisiert - abhängig von den jeweiligen Anwendererfordernissen - die Verwaltung der logischen und physischen Ein-/Ausgabegeräte wie V.24-Konsole, Floppy-Disk und Drucker mit vollkommen freizügiger Datenstromsteuerung durch den Nutzer.
Vom OS-Operationssystemkern wird außerdem die Abarbeitung von 11 internen UDOS-Kommandos übernommen:

- DEBUG Aufruf des U880-Softwaremonitors
- INITIALIZE Initialisierungsrequest
- BRIEF Briefmode, kein Kommandoecho auf Monitor

- XEQ nochmaliges Ausführen des letzten Kommandos
- VERBOSE Verbosemode, Kommandoecho auf Monitor
- ALLOCATE Besetzen eines Systemspeicherbereichs
- DEALLOCATE Freigabe eines Systemspeicherbereichs
- FORCE erzwungener Systemaufruf
- CLOSE Closerequest an eine logische Einheit
- : Hexadezimalrechnung

Dateiverwaltungssystem NDOS - Das Dateiverwaltungsprogrammsystem NDOS des Betriebssystems UDOS ist auf die Nutzung der Floppy-Disk als externen Massendatenspeicher zugeschnitten. Es organisiert die physische und logische Verwaltung der Dateien auf den System- und Anwenderdisketten.

UDOS-Systemkommandos - Während die bisher genannten UDOS-Programmteile fester Bestandteil des Betriebssystems sind, handelt es sich bei den Systemkommandos um nachladbare Softwarekomponenten, die dem Anwender den Zugriff zu den UDOS-Systemkomponenten sichern.

- ACTIVE Aktivierung eines Ein-/Ausgabegerätes
- CAT Ausgabe von Diskettеninhaltsverzeichnissen
- COMPARE Vergleich zweier Diskettendateien
- COPY Kopieren von Dateien
- COPY.DISK Kopieren von Disketten
- DATE Datumsein-/ausgabe
- DEACTIVATE Deaktivierung eines Ein-/Ausgabegerätes
- DEFINE Definieren einer logischen Einheit
- DELETE Löschen von Diskettendateien
- DO Ausführen einer UDOS-Kommandodatei
- DUMP Ausgabe eines Hexadezimaldump
- ECHO Ausgabe eines Operatortextes
- ERROR Erläuterung von Fehlerkodes
- ERRORS Diskettenzugriffsfehler seit Systemstart
- EXTRACT Ausgabe der Parameter einer Diskettendatei
- FORMAT Formatieren einer Diskette für UDOS
- HELP UDOS-Kommandoerläuterungen
- IMAGE Laden einer Diskettendatei
- LADT Zuordnung von Ein-/Ausgabegeräten
- MASTER Definition des Mastergerätes
- MOVE Transportieren einer Diskettendatei
- PAUSE Operatorreaktionspause
- PRINT Ausgabe einer Textdatei
- RENAME Umbenennen einer Diskettendatei
- SET UDOS-Parameter setzen
- SETFD Setzen der FD-Laufwerkskonfiguration
- SETLP Setzen der Druckerparameter
- STATUS Ausgabe des Diskettenstatus

4.2.2. UDOS-System- und Dienstprogramme

Die unter Steuerung des Betriebssystem UDOS laufenden Dienstprogramme helfen dem Anwender bei der Lösung seiner speziellen Aufgabenstellung (Erstellen von Programmen, Übersetzen von Programmen und deren Test u.a.m.).

- EDIT zeilenorientierter Editor

- UFORM	Textverarbeitungssystem
- ASM	U880-Makroassembler
- PLINK	Linker für U880-Programme
- U881ASM	U881/882-Assembler
- U8000ASM	U8000-Assembler
- LINK	Linker für U881/U882- und U8000-Programme
- IMAGER	Imager für U881/U882- und U8000-Programme
- RABUG	Fehlersuchprogramm für U880-Programme
- FILE.DEBUG	Diskettendebugger
- DISKTEST	Diskettentestprogramm
- SI	Treiber für serielles Interface
- remote	Kopplung von UDOS mit WEGA-Systemen
- LP	Druckertreiber für V.24 und IFSS
- UPROG	Programmierung von EPROM
- PLZ/ASM U880	PLZ/ASM-Compiler für U880
- PLZ/SYS U880	PLZ/SYS-Compiler für U880
- BASIC U880	BASIC-Interpreter für U880
- PASCAL U880	PASCAL-Compiler für U880
- FORTRAN U880	FORTRAN-Compiler für U880

4.3. OS/M Betriebssystem

Das Betriebssystem OS/M ist ein Floppy-Disk orientiertes Betriebssystem für den 8-Bit-Mikrorechnerteil des P8000. Die Anwendung von OS/M ist begründet in der Verfügbarkeit einer großen Anzahl von Dienstprogrammen, von Compilern/Interpretern für höhere Programmiersprachen und von Applikationssoftwaresystemen. Sie betreffen den Komplex der kommerziellen Datenverarbeitung, den Komplex der Mikrorechnerentwicklungssoftware, aber auch viele weitere Anwendunglinien.
Die Schnittstelle von OS/M auf dem P8000 zu den unter seiner Steuerung laufenden Programmen ist voll kompatibel mit der Schnittstelle des Betriebssystems CP/M Version 2.2. Außerdem ist die Kompatibilität mit dem in der DDR verfügbaren Betriebssystem SCP für die Mikrorechnersysteme A5120/30 und PC 1715 gegeben.
Mit zwei optional an das P8000 anschließbaren 8-Zoll-Standardlaufwerken wird die Dateiübernahme von CP/M-Standarddisketten einfacher Aufzeichnungsdichte (Standardformat 3740) gewährleistet.
Vom Betriebssystem OS/M werden standardmäßig folgende Ein-Ausgabegeräte des P8000 bedient:

- ein serieller V.24-Kanal für die OS/M-Systemkonsole
- ein bis maximal vier Floppy-Disk-Laufwerke 5 1/4 Zoll
- ein Drucker (Type: Epson LX86 oder robotron K6312 mit V.24-Schnittstelle)

Das Betriebssystem OS/M verwaltet einen Operationsspeicherbereich (RAM) von 64-KByte.

4.3.1. OS/M Struktur des Betriebssystems

Das Betriebssystem OS/M besteht aus drei Basis-Systemkomponenten:

- CCP Consol Command Processor
- BDOS Basic Disc Operating System
- BIOS Basic Input/Output System

Consol Command Processor (CCP) - Der CCP-Modul des Betriebssystems OS/M ist der Consol Command Processor. Er ist ein vollständig geräteunabhängiger Programmmodul, dessen Hauptaufgabe die Befehlsentschlüsselung der Operatoreingaben und die daraus folgende Einleitung bestimmter Maßnahmen ist.
Sechs resident im CCP geladene interne Befehle werden nach Erkennung sofort abgearbeitet.

DIR	Auslisten des Directoryverzeichnisses
ERA	Löschen einer Diskettendatei
REN	Umbenennen einer Diskettendatei
TYPE	Ausgabe von ASCII-Dateien auf die Systemkonsole
SAVE	Anlegen einer Speicherabzugsdatei (ab 100H)
USER	Einteilung einer Diskette in Benutzerbereiche

Basic Disc Operating System (BDOS) - Der BDOS-Modul des Betriebssystems OS/M übernimmt als zweiter geräteunabhängiger Teil des OS/M die gesamte Dateiverwaltung der auf Disketten befindlichen Datenbestände. Für alle im BDOS verankerten Funktionen - sie werden als BDOS-Systemaufrufe bezeichnet - existiert ein einheitliches Parameterübergabeschema.
Von den insgesamt 37 BDOS-Systemaufrufen beziehen sich 25 auf die Floppy-Disk-Einheit, die restlichen 12 auf die Systemkonsole, auf den Drucker und auf einen speziellen seriellen V.24-Ein-/Ausgabekanal.

Basic Input/Output System (BIOS) - Der BIOS-Modul stellt den gerätespezifischen Teil des Betriebssystem OS/M dar. In ihm sind alle Ein-/Ausgabetreiberprogramme zur Bedienung der Peripheriegeräte zusammengefaßt:

Die Aktivierung des Betriebssystems OS/M erfolgt über den EPROM-Monitor des P8000 (siehe auch Beriebssystem UDOS) der zum Einlesen des BIOS-Moduls benutzt wird. Anschließend wird der EPROM-Speicherbereich abgeschaltet und durch einen gleich großen RAM-Bereich ersetzt. Ein sogenannter Kaltstartlader und Warmstartlader im BIOS sorgen danach für das Einlesen des BIOS- und CCP-Moduls sowie für die OS/M-Systeminitialisierung.
Der Floppy-Disk-Treiber des Betriebssystems OS/M realisiert die Ansteuerung von:

5 1/4" Minilaufwerken 40 Spuren
(MFM 16 Sektoren je Spur a' 256 Byte)
5 1/4" Minilaufwerken 80 Spuren
(MFM 16 Sektoren je Spur a' 256 Byte)
5 1/4" Minilaufwerken 80 Spuren
(MFM 32 Sektoren je Spur a' 256 Byte)
8" Standardlaufwerken 77 Spuren
(FM 26 Sektoren je Spur a' 128 Byte)

Die Systemkonsole des Betriebssystems OS/M wird über eine V.24-SChnittstelle angeschlossen und bedient.
Außer den genannten Ein-/Ausgabegeräteprogrammen ist im OS/M ein Treiber für eine serielle Schnittstelle implementiert. Dieser ersetzt die Treiber für Lochbandstanzer und Lochbandleser. Er kann vom OS/M-Anwender des P8000 über die Systemnamen der Lochstreifen-Ein-/Ausgabegeräte angesprochen werden.

4.3.2. System- und Dienstprogramme OS/M

Zum Basissoftwaresystem OS/M des P8000 gehören folgende OS/M-Programme

ED	zeilenorientierter Texteditor
ASM	Assembler
DDT	Fehlersuchprogramm
LOAD	Konvertierung HEX-Format OS/M-Kode

STAT	Diskettenstatusermittlungsprogramm
PIP	Diskettendateitransferprogramm
DUMP	hexadezimale Ausgabe einer Maschinen-kodedate
SUBMIT/XSUB	Abarbeitung von OS/M-Kommandodateien
SYSGEN	Generierung einer neuen OS/M-Diskette
FORMAT	Formatierungsprogramm
GENHEX	Konvertierung OS/M-Kode in HEX-Format
COPYDISK	Kopieren von Disketten
ERRORS	Ausgabe der Diskettenfehlerstatistik
SETFD	Setzen der FD-Laufwerkskonfiguration
SETLD	Setzen der Druckerparameter

5. Echtzeitsoftwarekomponenten des Gerätesystems P8000

Echtzeitbetriebssysteme sind definitionsgemäß dadurch gekennzeichnet, daß der Rechner, auf dem sie abgearbeitet werden, auf echte, real existierende zeitkritische Ergebnisabläufe in seiner Umwelt reagieren muß. als ein entscheidendes Maß für die Leistungsfähigkeit eines Rechners mit einem Echtzeitbetriebssystem ist deshalb die Reaktionszeit anzusehen (im Mikrosekundenbereich), die beim Umschalten (task switch) von einem Prozeß (Programm) mit niedriger Priorität auf einen Prozeß (Programm) mit höherer Priorität (z.B. Uhrimpuls, Beendigung einer Ein-/Ausgabeaktivität, Alarmmeldung ...) vergeht.

5.1. IRTS 8000 Echtzeitbetriebssystem

IRTS (INTEGRABLE REAL TIME SOFTWARE) ist ein durch den Anwender konfigurierbares Softwaresystem mit Multitaskeigenschaften für Echtzeitanwendungen. Es besteht aus einem realitv kleinen Echtzeitsystemkern (Kernel), um den system- und anwendungsspezifische Ergänzungsmodule angeordnet werden. Der Systemkern organisiert die Steuerung und Synchronisation von multiblen Systemanforderungen in einer Echtzeitumgebung.
Er enthält die Echtzeitbasisfunktionen:

- Tasksteuerung
- Semaphoresteuerung
- Intertask-Kommunikation (Mailboxes)
- Echtzeituhr
- Memory Management

IRTS kann sowohl in einem RAM- als auch in einem EPROM-basierenden Anwendersystem eingesetzt werden. Es unterstützt die Mikroprozessoren der Familie U8000.
Die Systemkonfigurierung von IRTS erfolgt rechnergestützt mit dem Sprachprozessor ICL (IRTC Configuration Language).

Als typische Umschaltzeit vom aktuell bearbeiteten Prozeß auf einen Prozeß mit anderer Priorität (Taskwechsel) hat sich bei IRTS ein Wert von ca. 250 Mikrosekunden erwiesen. Beim eintreffen von externen Ereignissen (interrupts) findet kein Taskwechsel statt, so daß die Interruptserviceroutinen als Teil des aktuell laufenden Task abgearbeitet werden, sofern nicht Systemrufe einen Prioritätswechsel erzwingen.
Zur Installation von IRTS werden neben dem Mikroprozessorsystem (CPU, RAM- bzw. EPROM-Speicher) ein interruptfähiger Zeitgeberbaustein zur Realisierung der Echtzeituhr und bei Anwendung des IRTS-Debuggers ein interruptfähiger Zählerbaustein benötigt, mit dem der CPU-Status "Stack Memory Request" beim Single-Step-Betrieb gezählt werden kann. Zur Unterbringung des Systemkerns von IRTS ist ein Speicherbereich von ca. 4 KByte vorzusehen.
Als spezielle Beispielapplikationslösung existiert eine Implementation des Echtzeitbetriebssystems IRTS auf dem Programmier- und Entwicklungssystem P8000.

Das Konzept des Echtzeitsteuerprogrammsystems IRTS basiert auf den drei Basiselementen Tasks, Semaphores sowie Mailboxes und Messages.

Tasks werden verwendet, um die Arbeit des Prozessors in möglichst unabhängige Teilkomponenten aufteilen zu können, um eine optimale Zeitbilanz auf einem Prozessor zu erzielen. Eine Task besteht aus einem Programm zusammen mit einem bestimmten CPU-Zustand (Registerinhalte, Flag-Belegungen, etc.).
Die Aufteilung in möglichst unabhängige Operationen, denen einzelne Tasks zugeordnet werden, hat den Vorteil, daß das aus dem Zusammenwirken von Teilkomponenten bestehende Gesamtsystem leichter verstanden und behandelt werden kann - vor allem auch in Bezug auf den zeitlich (CPU-intern) rein sequentiellen, nach außen jedoch scheinbar parallelen, Programmablauf.
Wichtig für die Arbeitsweise von IRTS ist, daß alle Tasks nach dem Systemstart laufbereit sind. Die Task müssen sich in der Folgezeit selbsttätig mit Hilfe der Dienstleitungen des IRTS-Kern (systemcall) verwalten. Semaphores werden zur Sperr-Synchronisation der Aktion verschiedener Tasks (z.B. konkurrierende Tasks mit gemeinsamen Variablen) verwendet. Die Synchronisation mit Hilfe von Semaphores besteht dabei darin, eine zeitlich definierte Einordnung dieser Aktionen zu erreichen. Eine Semaphore ist eine Datenstruktur, die eine Aussage erlaubt, welche Tasks auf die Ausführung einer bestimmten Operation warten.
Eine ganz typische Anwendung finden Semaphores bei der Verhütung von Zugriffskonflikten unterschiedlicher Tasks auf eine gemeinsame Ressource (RAM-Bereiche mit gemeinsamen Daten, Drucker, etc.).
Mailboxes sind ein Werkzeug zur Abwicklung der Intertaskkommunikation. Angewendet werden Mailboxes, wenn eine Information von einer Task an eine andere Task übermittelt werden soll oder wenn eine Ereignissynchronisation zweier Task mit einem Austausch einer beliebigen Menge von Informationen erfolgen soll. Die Information selbst wird in einem besonderen Nachrichtenträger, dem sog. Messageobjekt (kurz: Message), transportiert. Eine Mailbox ist dabei der Ort, wo Messages für eine andere Task hinterlegt und abgeholt werden können. In IRTS existieren spezielle Operationen, die das Senden einer Message an eine Mailbox und das Empfangen einer Message aus einer Mailbox in vielfältiger Weise unterstützen.

5.1.1. IRTS 8000 Systemkern

Der IRTS-Kern ist der Basisblock jedes IRTS-Echtzeitsteuerprogrammsystems. Er enthält die folgenden Echtzeitfunktionen:

- Task Management
- Semaphores
- Clock Management
- Memory Management
- Intertask Communication

Alle Anforderungen an den IRTS-Kern erfolgen über einen Systemcall. Die Parameterübergabe vom Anwenderprogramm an den IRTS-Kern erfolgt in Registern.

Task Management - das ist der Mechanismus zur Steuerung und Überwachung des Ablaufes miteinander konkurrierender Operationen auf einem einzelnen Prozessor. Eine Task besteht aus dem Maschinenkode (Programm), dem Systemmode-Stack, wahlweise einem Normalmode-Stack, dem Taskobjekt (auch als Taskdatenstruktur bezeichnet - einem jeder Task zugeordneten IRTS-Informationsblock zur Taskzustandserfassung und zur Tasksteuerung)m dem Stackpointer, dem Pointer auf den nächsten auszuführenden Maschinenbefehl (Program Counter), dem CPU-Registerzustand und anderen Informationen.

Create Task	Erzeugung einer Task
Destroy Task	Redefinition einer Task
Reschedule Task	Änderung der Taskpriorität
Unlock Task	Freigabe des Taskwechsels
Lock Task	Blockierung des Taskwechsels
Suspend Task	Suspendierung einer Task
Resume Tsk	Wiederaktivierung einer Task
Wait Task	Selbstsuspendierung einer Task
Who I am	Bereitstellen der Taskobjektadresse
Census of the Task	Ermittlung von Taskinformationen

Semaphores - Semaphores dienen zur Synchronisation sich gegenseitig beeinflussender Tasks.
Eine Semaphore besteht aus einem Zähler, der die unbedienten Anforderungen (Signale) an eine Ressource aufsummiert und aus einer zugehörigen Warteschlange von Tasks, die auf diese Ressource zugreifen wollen.
Ein positiver Zahlwert des Zählers einer Semaphore zeigt an, daß die Ressource frei, d.h. verfügbar ist - der Zahlwert selbst, wie viele Anforderungen sofort gleichzeitig bedient werden könnten. Der Wert Null oder ein negativer Zahlwert des Zählers zeigen an, daß die Ressource nicht verfügbar ist und eine Anforderung in die Warteschlange eingereiht wird. Der negative Zahlwert selbst gibt Auskunft, wie viele Anforderungen (Signale) bereits in die Warteschlange eingereiht wurden. Jede Anforderung dekrementiert den Zähler einer Semaphore - jede bediente Anforderung inkrementiert ihn wieder.

Create Semaphore	Definition einer Semaphore
Destroy Semaphore	Redefinition einer Semaphore
Wait Semaphore	Warten auf ein Semaphorensignal
Signal Semaphore	Setzen eines Semaphorensignals
Test Semaphore	Semaphorenzustandsermittlung
Clear Semaphore	Auflösen einer Semaphore

Clockmanagement - IRTS arbeitet mit einer interruptgesteuerten Echtzeituhr. Die Zykluszeit dieser Uhr ist abhängig von der gerätetechnischen Realisierung der Interruptquelle. Die interne Software-Uhr wird verwendet für zeitabhängiges Warten von Tasks (timed waits), für zeitabhängiges Aussetzen von Tasks (timeout) und für die zyklisch wechselnde Abarbeitung von Tasks (round robin scheduling).

Alle Zeiteinheiten in IRTS werden in 'ticks' (Uhrimpulsen) gemessen und behandelt.

Set Clock	Setzen der Echtzeituhr
Read Clock	Lesen der Echtzeituhr
Clk_Delay_Absolute	Clock-Warteschlange absolut
Clk_Delay_Interval	Clock-Warteschlange relativ

Memory Management - Das Echtzeitsteuerprogrammsystem IRTS beinhaltet Funktionen zur Verwaltung eines gemeinsamen RAM-Speicherpools für Programme und Datenstrukturen. Diese IRTS Memory Management Funktionen ermöglichen die dynamische Belegung und Freigabe von Speicherbereichen durch die Anwendertasks.

Allocate Memory	Belegung eines Speicherbereiches
Release Memory	Freigabe eines Speicherbereiches
Memory Census	Ermittlung des IRTS-Speichersatus

Intertaskkommunikation - Die Intertask-Kommunikationsfunktionen von IRTS ermöglichen den Austausch von Informationspaketen (Messages) zwischen unterschiedlichen Tasks. Der Kommunikationsprozeß zwischen zwei Tasks wird in das Senden einer Message durch die eine Task und in das Empfangen einer Message durch die andere Task aufgeteilt. Der Ort, an dem die Message von einer Task hinterlegt wird und eine andere Task auf sie zugreifen kann, wird als Mailbox (Briefkasten) bezeichnet.

Create Mailbox	Definition einer Mailbox
Destroy Mailbox	Freigabe einer Mailbox
Create Message	Definition einer Message
Destroy Message	Freigabe einer Message
Acquire Message	Zuweisung einer Message
Assign Message	Zuweisung initialisierte Message
Send Message	Senden einer Message
Reply Message	Rücksenden einer Message
Receive Message	Empfangen einer Message
Release Message	Freigabe einer Message
Get Message	Ermittlung von Statusinformationen
Read Message	Lesen einer Message
Write Message	Verändern einer Message

5.1.2. IRTS 8000 Konfigurationssprache ICL

Durch den streng modularen Aufbau von IRTS ist eine sehr dynamische Anpassung an ganz verschiedenartige Anwendungskonfigurationen möglich. Die Forderung nach einer automatisierten Systemgenerierung wird durch die Verfügbarkeit eines Konfigurations-Sprachprozessors (ICL) erfüllt. Er erlaubt die Verarbeitung von High-Level-Generierungssprachelementen für die notwendigen Hardware-Informationen, für Softwareparameter, für die Linkage-Informationen und für die Systemdatenstruktur.

Die Hauptelemente der ICL-Sprache sind:

CONSTANTS	Spezifizierung von Systemkonstanten
EXCHANGES	Definition einer applikationsmailboxes
FILES	Konfigurations-Link-Dateien
HARDWARE	Beschreibung der Applikationshardware
INITIALIZATION	Spezifizierung der Initialisierungsroutine
INTERRUPT	Definition der Interrupt-Bedingungen
MEMORY	Definition der Speicherkonfiguration
SECTIONS	Spezielle Sektoraufteilungsinformationen
SEMAPHORES	Definition der Applikationssemaphoren
SWITCHES	Flags für den Systemgenerierungsprozess
TASKS	Definition der Applikationstasks

5.1.3. IRTS 8000 Ausgabeorganisation PRINTF

Die Verwendung der IRTS-Ausgabeorganisation ist insbesondere für Applikationen auf Assemblerniveau immer dann zu empfehlen, wenn die Kommunikationssoftware übersichtlich, änderungsfreundlich oder projektspezifisch generierbar sein soll.
Das Prinzip der Arbeitsweise der Ausgabeorganisation besteht darin, daß eine angegebene "Format"-Anweisung interpretativ abgearbeitet wird und die entstehenden Zeichenketten an das angegebene Gerät weitergereicht werden.
Eine Formatanweisung besteht im Normalfall aus einer Zeichenkette mit direkt auszugebenden Texten und darin eingelagerte Steueranweisungen.

5.1.4. IRTS 8000 Terminal- und Drucker-Handler

Der IRTS-Terminal-Handler besteht aus den zwei weitgehend unabhängigen Teilkomponenten Eingabe-Handler (CONIN-Handler) und Ausgabe-Handler (CONOUT-Handler).
Der CONIN-Handler realisiert die Eingabe von Informationen über ein serielles Interface. Der Handler gestattet neben der normalen Eingabe von Zeichen die Eingabe von Cursor-Steuerfunktionen sowie ESCAPE- und Control-Zeichen. Auf jedem Terminal können dabei quasiparallele Ausgaben in den folgenden drei Arten vorgenommen werden, wobei wahlweise für jede Ausgabeart ein besonderer Bildausschnitt (Window) definier werden kann:

1) Ausgabe des aktuellen Puffers der Tastatureingabe, wobei der Cursor unabhängig von weiteren Ausgaben auf dem Terminal, bis zum Eingabeende auf der nächsten Eingabeposition verharrt.
2) Ausgabe von Vorrangmeldungen.
3) Ausgabe von Informationen, wobei die laufende Ausgabe jederzeit durch die Eingabe von XOFF (Control-S) unterbrochen und durch XON (Control-Q) fortgesetzt werden kann.

Der Drucker-Handler realisiert alle Druckerausgaben unter IRTS. Der Handler selbst wird als eine oder mehrere Task(s) verwaltet, wobei das Prioritätsniveau der Task(s)

vom Anwender festgelegt wird. Die Ausgabe aus einem Anwenderprogramm auf einen Drucker erfolgt über den IRTS-System-Call M_SEND, mit dem ein Nachrichtenträger (Message), der die auszugebende Information transportiert und mit einem speziellen Nachrichtenkode (Request) versehen ist, an die dem Gerät vom Anwender zugeordnete Mailbox versendet wird. Dabei wird mit dem Request festgelegt, ob die Ausgabe asyncron oder synchron (im Wait-Betrieb) erfolgen soll.

5.1.5. IRTS 8000 Testhilfsmittel DEBUGGER/MONITOR

Der IRTS-Debugger ist ein Testhilfsmittel für Anwenderprogramme, die unter der Steuerung von IRTS laufen sollen und deren Echtzeitumgebung während des Testbetriebs weitgehend erhalten bleiben muß. Das bedeutet, daß während des Testbetriebes sowohl die Echtzeituhr als auch bereits betriebsfähige Anwenderprogramme parallel weiterlaufen.
Der Debugger meldet sich nach RESET im Standalone-Mode. In diesem Mode können Programme vom Host-System nachgeladen werden und anwendereigene Initialisierungsroutinen ausgetestet werden:

BREAK-Kommando	Softwarehaltepunkt - das zu testende Programm muß in einem RAM-Speicherbereich geladen sein. Der Debugger verwaltet im Standalone-Mode einen, sonst eine beliebige Anzahl von Haltepunkten. Mit dem Erreichen eines Haltepunktes werden etwaige Zeitaktivitäten eines Programms eingefroren.
GO-Kommando	Das GO-Kommando bewirkt die Fortsetzung des unterbrochenen Programms mit dem aktuellen Stand des Programmzählers. Wurde in der Zeitspanne,in der sich das zuletzt unterbrochene Programm unter der Steuerung des Debuggers befand, ein oder mehrere Haltepunkte angelaufen, so wird die älteste Unterbrechungsmeldung auf der Konsole angezeigt.
REGISTER-Kommando	Das REGISTER-Kommando dient zur Anzeige und Veränderung von Speicherinhalten, die der unterbrochenen Task zugeordnet sind. Der Inhalt des FCW wird dabei in Mnemoniks angegeben.
DISPLAY-Kommando	Das DISPLAY-Kommando dient zur Anzeige und Veränderung von Speicherinhalten. Die Anzeige kann wahlweise im Byte-, word- oder Longformat erfolgen.
TRACE/NEXT-Kommando	Unabhängig von weiteren Aktivitäten des gesamten Programmsystems kann ein einzelnes Anwenderprogramm nach dem Anlaufen eines Haltepunkts im Einzelschrittbetrieb durchfahren werden. Im TRACE werden alle Zwischenschritte - ausgenommen der Aktivitäten des Kerns

- protokolliert. Am Ende des Kommandos werden die aktuellen Registerinhalte angezeigt.

Im IRTS- Debugger sind außerdem Kommandos zum Lesen/Schreiben von Ports und zum Füllen, Vergleichen und Verschieben sowie zum Laden vom/zum Hostsystem von Speicher-Inhalten implementiert.

Der IRTS-Monitor ist ein Ergänzungsmodul des IRTS-Debuggers. Er dient in erster Linie der Anzeige von aktuellen Zuständen eines unter IRTS laufenden Anwendersystems:

System-Visite	Je nach Modifikation des Kommandos können der Zustand des Anwendersystems, der Zustand der Speicher-Verwaltung, der Inhalt des Program Status Area und die Systemlaufzeit zur Anzeige gebracht werden.
Task-Historie	Es werden die letzten 16 (beim U8002: 32) Tasks in der Reihenfolge ihrer Bearbeitung aufgelistet.
Task-Visite	Je nach Modifikation des Kommandos werden alle generierten Tasks, alle laufbereiten Tasks, alle zeitverwalteten Tasks, alle inaktiven Tasks, alle relevanten Informationen einer Task oder die Registerinhalte einer Task zum Zeitpunkt der letzten Unterbrechung angezeigt.
Task-Handliong	Mit dem Taskshandling können Task generiert und zerstört, gestoppt und fortgesetzt werden.
Mailbox-Visite	Je nach Modifikation können alle statisch generierten Mailboxes aufgelistet werden, oder die Auslistung relevanter Parameter der Mailbox sowie die Informationsinhalte der in einer angewählten Mailbox enthaltenen Messages zur Anzeige gebracht werden.
Mailbox-Handling	Mit dem Kommando zum Mailboxhandling können Mailboxes generiert oder zerstört werden; ferner kann an jeweils eine Mailbox eine Nachricht versandt werden.
Semaphore-Visite	Je nach Modifikation können alle statisch generierten Semaphores zur Anzeige gebracht werden oder der Inhalt der Warteschlange einer Semaphore ausgelistet werden.
Semaphore-Handling	Mit dem Kommando zum Semaphorehandling können Semaphores generiert, zerstört oder deren Initialwert neu festgesetzt werden.

ANHANG

Anhang A: Fehlerliste Hardwareeigentest P8000-Grundgerät

Fehler-nummer	Fehlerparameter und Fehlerbeschreibung Parameter1	Parameter2	Parameter3	Parameter4
	* * * 8-Bit-Mikrorechnerteil * * *			
Test EPROM				
00	-	-	-	-
	EPROM-Prüfsummenfehler			
Test statischer RAM				
05	ADR	TD	RD	-
	Speicheradresse fehlerhaft			
06	ADR	TD	RD	-
	Datenleitung fehlerhaft			
07	ADR	TD	RD	-
	Durchschieben von 0 oder 1 fehlerhaft			
08	ADR	-	-	-
	M1-Test fehlerhaft			
Test PIO				
10	PIO PORT #	TD	RD	-
	Daten 'AA' oder '55' fehlerhaft			
Test CTC				
15	CTC PORT #	TD	RD	-
	Fehler im Zeitkonstantenregister			
16	CTC PORT #	TD	RD	-
	Kanal zählt nicht			
17	CTC PORT #	-	-	-
	Kanal gibt keinen Interrupt			
Test SIO				
20	SIO PORT #	TD	RD	-
	Fehler im Interruptregister			
21	CTC PORT #	-	-	-
	Kanal gibt keinen Interrupt			
Test FDC				
25	FDC MSR	-	RD	-
	FDC-Fehler keine Bereitmeldung Kommandoeingabe			
Test DMA				
26	DMA ADR	-	RD	-
	Statusbyte fehlerhaft			
27	DMA ADR	-	-	-
	DMA gibt keinen Interrupt			

```
Test dynamischer RAM
30      ADR         TD          RD          -
        Speicheradresse fehlerhaft
31      ADR         TD          RD          -
        Datenleitung fehlerhaft
32      ADR         TD          RD          -
        Durchschieben von 0 oder 1 fehlerhaft
33      ADR         -           -           -
        M1-Test fehlerhaft

------------------------------------------------------------

          * * *  16-Bit-Mikrorechnerteil  * * *


Test EPROM
40 (*)  -           -           -           -
        EPROM-Prüfsummenfehler

Test statischer RAM
46      ADR         TD          RD          -
        Datenleitung fehlerhaft
47      ADR         TD          RD          -
        Daten 'AAAA' fehlerhaft
48      ADR         TD          RD          -
        Daten '5555' fehlerhaft

Test PIO
50      PIO PORT #  TD          RD          -
        Daten 'AA' oder '55' fehlerhaft

Test CTC
55      CTC PORT #  TD          RD          -
        Fehler Daten 'AA' oder '55' Zeitkonstantenreg.
56      CTC PORT #  -           -           -
        Kanal zählt nicht
57      CTC PORT #  -           -           -
        Kanal gibt keinen Interrupt

Test SIO
60      SIO PORT #  TD          RD          -
        Fehler 'AA' oder '55' im Interruptregister
61      CTC PORT #  -           -           -
        Kanal gibt keinen Interrupt

Test dynamischer RAM
70 (*)  -           -           -           -
        kein DRAM-Speicher
71 (*)  SEG #       ADR         RD          -
        Segmentadresse fehlerhaft
72      SEG #       ADR         TD          RD
        Speicheradresse fehlerhaft
73      SEG #       ADR         TD          RD
        Datenleitung fehlerhaft
74      SEG #       ADR         TD          RD
        Daten 'AAAA' fehlerhaft
75      SEG #       ADR         TD          RD
        Daten '5555' fehlerhaft
```

76 (*) - - - -
keine fehlerfreien Segmente oberhalb Segment 0

Test MMU
80 MMU PORT # SDR FELD # TD RD
MMU's sind nicht einzeln adressierbar
81 MMU PORT # SDR FELD # TD RD
SAR oder DSCR Indizierung ist fehlerhaft
82 MMU PORT # SDR FELD # TD RD
SDR Daten 'AA' oder '55' sind fehlerhaft
83 MMU CMD # TD RD -
MMU Controlregister Fehler 'AA' oder '55'
84 REG # TD RD -
System-/Normal-Break Fehler 'AA' oder '55'
85 MMU ID # SDR # VDAT -
Stack-MMU gibt keinen Trap beim Limit-Test
86 MMU ID # SDR # VDAT -
Unerwarteter Trap
87 MMU ID # SDR # VDAT -
Unerwarteter Trap
88 MMU ID # SDR # VDAT -
Daten-MMU gibt keinen Trap beim Limit-Test
89 MMU ID # SDR # VDAT -
Stack MMU-gibt keinen Trap beim Read-Only-Test
90 MMU ID # SDR # VDAT -
Daten MMU-gibt keinen Trap beim Read-Only-Test
91 MMU PORT # SDR # TD RD
Übersetzung der Daten-MMU ist fehlerhaft
92 MMU PORT # SDR # VDAT -
Unerwarteter Trap bei Testschritt 91
93 MMU PORT # SDR # TD RD
Übersetzung der Stack-MMU ist fehlerhaft
94 MMU PORT # SDR # VDAT -
Unerwarteter Trap bei Testschritt 93
95 MMU PORT # SDR # TD RD
Übersetzung der Kode-MMU ist fehlerhaft
96 MMU PORT # SDR # VDAT -
Unerwarteter Trap bei Testschritt 95
97 MMU PORT # SDR # - -
kein Trap beim Limit-Test der Kode-MMU

Zeichenerläuterung:

(*) verhängnisvoller Fehler (FATAL ERROR), der zu einem Abbruch des Hardwareeigentests führt

Testparameter der Fehlermeldungen:
PIO PORT # -- Portadresse des getesteten PIO-Kanals
CTC PORT # -- Portadresse des getesteten CTC-Kanals
SIO PORT # -- Portadresse des getesteten SIO-Kanals
SEG # -- Segmentnummer
ADR -- Offsetadresse
TD -- Testdatenwert
RD -- Rückgelesener Datenwert
FDC MSR -- Adresse Hauptstatusregister FDC

DMA ADR -- Adresse DMA
MMU PORT # -- MMU-Portadresse der getesteten MMU
MMU CMD # -- MMU-Portadresse geodert im High-Teil mit dem entsprechenden MMU-control-Register-Kommando (00-Mode-Reg. / 01-SAR / 02-DSCR)
SDR FELD # -- relativer Zeiger (0-255) auf ein Byte eines SDR (64 SDR zu je 4 Byte, d.h. 256 Byte)
REG # -- Portadresse des getesteten Break-Registers
MMU ID # -- MMU-Identifier bei Rückkehr von einem Segmenttrap
SDR # -- logische Segmentnummer bzw. ein Satz von SDR (0-63)
VDAT Daten einer Speicherzugriffsverletzung bei einem einfachen MMU-Trap
High-Byte: Datenwert des Bus-Cycle-Status-Registers
Low-Byte: Datenwert des Violation-Type-Registers

(Hinweis: Die Übersicht der Portadressen des 8- und 16-Bit-Mikrorechnerteils befindet sich im P8000 Hardware-Handbuch)

Anhang B: U880-Monitorbeschreibung P8000-Grundgerät

Das Testen von Programmen wird durch den U880-Softwaremonitor unterstützt. Dazu gehören vor allem die Kommandos "B" (Break), "N" (Next), "G" (Go), "GE" (Get) und "S" (Save).
Unterbrechungspunkte im Anwenderprogramm werden durch das eintragen des Befehlskodes 0FFH (Restart 38) an Stelle des Programmbefehlskodes erzeugt. Die im U880-Softwaremonitor auf der Adresse für dem Restart 38 vorhandene Programmfolge speichert die aktuellen Register-, Programmzähler- und Stackpointerwerte des Anwenderprogramms und zeigt die Adresse der Programmunterbrechung mit "BREAK AT XXXX" an. Das hat zur Folge, daß der Befehlskode 0FFH (auch im Betriebssystem UDOS) nicht eingesetzt werden kann, da er in jedem Fall einen Unterbrechungspunkt erzeugt.
Ein Unterbrechungspunkt kann auch durch fehlerhafte Programmabläufe bei der Auswertung des 2. oder 3. Bytes eines Befehls (mit 0FFH als Inhalt) entstehen.
Manuell kann eine beliebige Anzahl von Unterbrechungspunkten im Anwenderprogramm eingetragen werden, wobei der ursprüngliche Befehlskode hier auch manuell wieder eingetragen werden muß.
Das Kommando "B" setzt einen Unterbrechungspunkt und merkt sich die Unterbrechungsadresse sowie den ursprünglichen Befehlskode. Durch das Kommando "B" ohne Parameter oder die Eingabe eines neuen Unterbrechungspunktes wird der vorherige Unterbrechungspunkt durch Eintragen des ursprünglichen Befehlskodes automatisch gelöscht.
Mit "Go" oder "Next" kann ein Programm an einem Unterbrechungspunkt fortgesetzt werden, ohne daß dieser gelöscht werden muß.
Bei der Arbeit mit Unterbrechungspunkten ist zu beachten, daß eine Befehlskodemanipulation ausgeführt wird. Das zu testende Programm darf deshalb nicht im EPROM-Speicherbereich stehen. Nach Erreichen des Unterbrechungspunktes tritt eine Zeitverzerrung auf, was bei der Programmtestung zu beachten ist.
Für die Arbeit mit dem Next-Kommando ist folgendes zu berücksichtigen:

- Das Kommando "N" arbeitet unter Nutzung eines CTC-Interrupts
- Das zu testende Programm darf nicht den Kanal 2 des CTC0 benutzen.
- Die zu testende Programmfolge darf nicht innerhalb einer Interruptserviceroutine der Kanäle 0 oder 1 des CTC0 stehen.

Das Kommando "GE" (Get) ermöglicht das Laden von Programmen als UDOS-Datei von Disketten ohne Start des Betriebssystems UDOS.
Das Kommando "S" (Save) ermöglicht das Rückladen von Programmen als UDOS-Datei auf Disketten ohne Start des Betriebssystems selbst.

Kommandoübersicht

In der Kommandobeschreibung werden folgende Vereinbarungen verwendet:

( ) Kennzeichen von optionalen Parametern
/ steht für logisches ODER

Für die Eingabe der Kommandos gilt:

- alle Kommandos und Parameter können in Klein- und Großbuchstaben geschrieben werden
- Zahlen müssen hexadezimal geschrieben werden und können aber mit Buchstaben beginnen
- fehlerhafte Eingaben (falsche Kommandos, falsche Parameter) werden mit dem Fehlerkennzeichen '?' quittiert

D	Display	adr (datenzahl)
F	Fill	anfangsadr endadr datenbyte
M	Move	quelladr zieladr byteanzahl
PR	Port Read	portadr
PW	Port Write	portadr datenbyte
R	Register	(registername)
B	Break	(adr)
G	Go	(startadr)
N	Next	(befehlsanzahl)
GE	Get	dateiname
S	Save	dateiname anfangsadr endadr (E=entry_point)(RL=record_length)
Q	Quit	
T	Test	
O	Operating System	
X	Execute	

Kommando "D" (Display) - Anzeigen und Verändern von Speicherinhalten:

D adresse (datenanzahl)

adresse: Anfangsadresse des anzuzeigenden bzw. zu verändernden Speicherbereiches
datenanzahl: Anzahl der anzuzeigenden Datenbytes
Ohne Angabe der optionalen Datenanzahl wird nur ein Datenbyte angezeigt, dessen Wert verändert werden kann. Folgende Eingaben können erfolgen:
datenbyte: Datenbyte wird in den Speicher geschrieben und Übergang zu nachfolgender Speicheradresse
"_" : Übergang zu vorangehender Speicheradresse
CR : Übergang zu nachfolgender Speicheradresse
Q : Verlassen der Display-Routine

Kommando "F" (Fill) - Füllen eines Speicherebreiches mit einem Datenbyte:

F anfangsadresse endadresse datenbyte

anfangsadresse: Anfangsadresse des Speicherbereiches
endadresse: Endadresse des Speicherbereiches
datenbyte: Datenbyte, mit dem der Speicher gefüllt werden soll

---

Kommando "M" (Move) - Verschieben eines Speicherbereiches:

M quellanfangsadresse zielanfangsadresse byteanzahl

quellanfangsadresse: Anfangsadresse des zu verschiebenden Speicherbereiches
zielanfangsadresse: Anfangsadresse des Zielbereiches
byteanzahl: Anzahl der zu verschiebenden Bytes

Kommando "PR" (Port Read) - Dateneingabe (Lesen) von einem Port:

PR portadresse

portadresse: Adresse des zu lesenden Ports

Kommando "PW" (Port Write) - Datenausgabe (Schreiben) an einen Port:

PW portadresse datenbyte

portadresse: Adresse des zu schreibenden Ports
datenbyte: an den Port auszugebendes Byte

Kommando "R" (Register) - Anzeigen und Verändern von Registerinhalten:

R (registerbezeichnung)

registerezeichnung:
A/B/C/D/E/F/H/L/I/A'/B'/C'/D'/E'/F'/H'/L'/IX/IY/PC/SP
Nach der anzeige des Inhaltes des angegebenen Registers kann der Registerinhalte verändert werden.

datenwert: Register wird mit dem Datenwert beschrieben und Übergang zu nachfolgendem Register
CR: Übergang zu nachfolgendem Register
Q: Verlassen der Registerroutine

Fehlt die Anzeige einer Registerbezeichnung, so erfolgt die Anzeige aller Registerinhalte.

Kommando "B" (Break) - Setzen und Löschen eines Unterbrechungspunktes:

B (adresse)

adresse: Adresse, auf die der Unterbrechungspunkt gesetzt werden soll (muß eine RAM-Adresse sein)
Wird keine Adresse angegeben, erfolgt ein Löschen des zuletzt eingegebenen Unterbrechungspunktes.

Kommando "G" (Go) - Programmabarbeitung starten oder fortsetzen:

G (startadresse)

startadresse: Adresse, bei der die Programmabarbeitung gestartet werden soll
Ohne Angabe der Startadresse wird die Programmabarbeitung beim aktuellen Programmzählerstand fortgesetzt.

Kommando "N" (Next) - Schrittweise Programmabarbeitung:

N (befehlsanzahl)

befehlsanzahl: Anzahl der im Einzelschritt abzuarbeitenden Befehle (ohne Angabe: 1 - maximal 256)
Nach abarbeitung eines jeden Befehls werden alle Registerinhalte ausgegeben.

Kommando "GE" (Get) - Laden einer UDOS-Maschinenkodedatei (Typ Procedure) von Diskette in den RAM des 8-Bit-Teils des P8000:

GE dateiname

dateiname: Name der zu landenden UDOS-Datei
Ohne Angabe der Laufwerksnummer wird die angegebene Datei auf der Diskette im Laufwerk 0 gesucht und entsprechend der im Deskriptor festgelegten Adressen geladen. Befindet sich die Diskette mit der angegebenen Datei in einem anderen Laufwerk, so muß die Laufwerksnummer entsprechend UDOS-Vorschrift im Dateinamen angegeben werden. Das Programm wird nicht geladen, wenn sich die Anfangsadresse im Adressenbereich befindet, der vom U880-Softwaremonitor selbst belegt wird. Folgende Fehlermeldungen werden von der Get-Routine erzeugt:

FILE ERROR:	die angegebene Datei konnte auf der Diskette im angegebenen Laufwerke nicht gefunden werden
DISK ERROR:	beim Laden der Datei trat ein Fehler beim diskettenzugriff auf
MEMORY PROTECTION:	die Anfangsadresse liegt im Speicherbereich, der durch den U880-Softwaremonitor belegt wird

Zur Beachtung: die Recordlänge der zu ladenden Datei darf nicht größer als 400H sein.

Kommando "S" (Save) - Auslagern eines Speicherausschnittes des 8-Bit-Teils des P8000 als UDOS-Maschinenkodedatei auf die Diskette:

S dateiname anfangsadresse endadresse
(E=entry_point)(RL=record_length)

dateiname:	Name der auszulagernden UDOS-Datei
anfangsadresse:	Anfangsadresse des auszulagernden Speicherbereiches
endadresse:	Endadresse des auszulagernden Speicherbereiches
entry_point:	Eintrittspunkt, der in den Deskriptor der UDOS-Datei eingetragen werden soll

record_length: Record-Länge der UDOS-Datei (implizit 100H);sie darf nicht größer als 400H sein

Folgende Fehlermeldung wird von der SAVE-Routine erzeugt:

DISK ERROR: beim Speichern der Datei trat ein Fehler beim Diskettenzugriff auf

Zur Beachtung: Nach Benutzung der SAVE-Routine und Rückkehr in das Betriebssystem UDOS muß unbedingt vor der Weiterarbeit das Kommando I gegeben werden. Durch die SAVE-Routine wurde der Diskettenbelegungsplan verändert und muß somit neu in den NDOS-Speicherbereich eingetragen werden.

Kommando "Q" (Quit) - Rückkehr in das Betriebssystem UDOS (wenn es vorher gestartet war):

Q

Das Kommando "Q" führt die Rückkehr in das Betriebssystem UDOS nicht aus, wenn vorher das Kommando "T" gegeben wurde. In diesem Fall muß das Betriebssystem neu gestartet werden.

Kommando "T" (Test) - Ausführung des Hardwareeigentests des 8-bit-Teils des P8000:

T

(siehe auch Abschn. 3.1.1.)

Kommando "O" (Operating System) - Start eines der 8-Bit- oder 16-Bit-Betriebssysteme:

O

Entsprechend eingelegter Systemdiskette wird der Start eines der Betriebssysteme vollzogen. Dazu wird für alle Betriebssysteme der Sektor 1, Spur 0 im Laufwerk 0 gelesen, Folgende Fehlermeldungen können auftreten:

DISK ERROR: beim Diskettenzugriff auf Spur 0, Sektor 1, Laufwerk 0 trat ein Fehler auf

INSERT SYSTEMDISK: die im Laufwerk 0 eingelegte Diskette ist keine Systemdiskette

Beim Laden des Betriebssystems UDOS können zusätzliche Fehlerausschriften bei fehlerhaftem Laden des OS oder NDOS mit anschließendem Rücksprung in den U880-Softwaremonitor auftreten (siehe auch UDOS-Systemhandbuch Abschn. 3.3.).

Kommando "X" (Execute) - Übergang in den U8000-Softwaremonitor:

X

(siehe Abschn. 3.1.2.2.).

System-Parameter U880-Softwaremonitor

Die folgenden Systemparameter sind für den Nutzer zugämglich:

NULLCT: Null Count (0FBFH)
Anzahl der Nullen, die nach einem "Line Feed" auf dem Terminal ausgegeben werden, initialisiert mit 00.

LFCNT: Line Feed Counter (0FC0H)
Anzahl der Line Feed's, die nach einem Wagenrücklauf ausgegeben werden, initialisiert mit 01.

PROMPT: Prompt-Zeichen (0FC1H)
ASCII-Kode des Zeichens, mit dem sich der U880-Softwaremonitor auf dem Bildschirm meldet, initialisiert mit 3EH.

LINDEL: Line Delete (0FC2H)
ASCII-Kode des Zeichens, welches das Löschen einer Zeile veranlaßt, initialisiert mit 7FH.

CHRDEL: Character Delete (0FC3H)
ASCII-Kode des Zeichens, welches das Löschen eines Zeichens veranlaßt, initialisiert mit 08H

STATQ: Statusbyte für Control-S, -Q und Escape (0FE0H).
Damit wird angegeben, ob Control-S, -Q und Escape wirksam sind. Es gilt:
- Bit 0 = 0 wirksam
- Bit 0 = 1 nicht wirksam

Das Statusbyte wird mit 10H initialisiert.

Sprungverteiler im U880-Softwaremonitor

Der im U880-Softwaremonitor enthaltene Sprungverteiler erlaubt dem Anwender den Zugriff auf vielbenutzte Monitorfunktionen durch einen absoluten Sprung zu der ausgewählten Routine. Es gehören dazu die Einzelzeichen-ein-/Ausgabe vom/zum Systemterminal, READY-Test der Floppy-Disk-Laufwerke und die Diskettentreiberroutine.
Im Einzelnen handelt es sich um folgende Funktionen:

GETA (Adresse im U880-Softwaremonitor: 0BE8H)
Diese Routine liest ein Zeichen aus dem Terminal-Interrupteingabepuffer. In der Routine wird nicht auf das Eintreffen eines Zeichens gewartet. Folgende Zusammenhänge sind gültig:
es ist ein Zeichen eingetroffen: -im Akkumulator steht das eingegangene Zeichen
- ZERO-Flag wird auf 1 gesetzt
es ist kein Zeichen eingetroffen:-der Akkumulator ist unverändert
- ZERO-Flag wird auf 0 gesetzt

PUTA (Adresse im U880-Softwaremonitor: 0BEBH)
Das im Akkumulator enthaltene Zeichen wird an das Systemterminal ausgegeben. Die Ausgabe kann durch die Eingabe von CONTROL S (ASCII-Kode:13H) unterbrochen werden. Die Eingabe von CONTROL Q (ASCII-KODE:11H) hebt CONTROL S auf.
Durch Veränderung des Statusbytes kann die Wirkung von CONTROL S und CONTROL Q aufgehoben werden.

PRESS (Adresse im U880-Softwaremonitor: 0BE5H)
Es wird abgefragt, ob vom Terminal ein Zeichen eingetroffen ist. Es gilt:
Zeichen eingetroffen: ZERO-Flag ist 1
Zeichen noch nicht eingetroffen: ZERO-Flag ist 0
Der Inhalt des Akkumulators bleibt unverändert.

PCON (Adresse im U880-Softwaremonitor: 0BEEH)
Der im U880-Softwaremontior enthaltene Konsoltreiber realisiert Zeicheneingaben und -ausgaben vom/zum Systemterminal entsprechend den im IY-Vektor übergebenen Angaben (Aufbau des IY-Vektors siehe DOK 6, Abschn. 6.2.). Folgende Requests werden realisiert:

READ ASCII (0CH)
Es werden solange Zeichen vom Systemterminal erwartet, eingelesen, gespeichert (entsprechend Adresse im IY-Vektor) und an das Systemterminal ausgegeben, bis die im IY-Vektor eingetragene Anzahl 0 ist oder ein Wagen-

rücklauf auftritt.
READ BINARY (0AH)
Eine im IY-Vektor eingetragene Anzahl von Zeichen wird vom Systemterminal erwartet, eingelesen und gespeichert.
WRITE ASCII (10H)
Es werden solange Zeichen aus einem Speicherbereich (entsprechend Adresse im IY-Vektor) an das Systemterminal ausgegeben, bis ein Wagenrücklauf auftritt oder die im IY-Vektor eingetragene Anzahl 0 ist.
WRITE BINARY (0EH)
Eine im IY-Vektor eingetragene Anzahl von Zeichen wird aus einem im IY-Vektor vorgegebenen Speicherbereich an das Systemterminal ausgegeben.

BTOHE (Adresse im U880-Softwaremonitor: 0BDFH)
Eine im Akkumulator befindliche 8-Bit binärcodierte Zahl wird in eine Hexadezimalzahl umgewandelt und entsprechend (HL) im RAM gespeichert.

OUTAS (Adresse im U880-Softwaremonitor: 0BF1H)
Initialisierung des IY-Vektors für den im U880-Softwaremonitor enthaltenen Konsoltreiber entsprechend (HL) für die Ausgabe von maximal 127 Zeichen an das Systemterminal.
Es gilt:

(HL)	Datenlänge, maximal 128
(HL+1)	Datenadresse, niederwertiger Teil
(HL+2)	Datenadresse, höherwertiger Teil

SSIGN (Adresse im U880-Softwaremonitor: 0BF4H)
Die Routine sucht aus dem vorgegebenen Eingabepufferspeicher das nächste Zeichen, das ungleich Zwischenraum (Kodierung 20H) ist. Der Zeiger für den Eingabepufferspeicher INPTR (Adresse im U880-Softwaremonitor: 0FBBH) zeigt auf das erste Zeichen ungleich Zwischenraum.

BDEC (Adresse im U880-Softwaremonitor: 0BF7H)
Es erfolgt der Rücksprung in den U880-Softwaremonitor ohne ausgabe des PROMPT-Zeichens. Das nächste Zeichen wird vom Systemterminal eingelesen und vom Kommandointerpreter ausgewertet.

DEBUG (Adresse im U880-Softwaremonitor: 0BFAH)
Es erfolgt der Rücksprung in den U880-Softwaremonitor mit Ausgabe des PROMPT-Zeichens. Das nächste über das Systemterminal eingegebene Zeichen wird durch den Kommandointerpreter ausgewertet.

READY (Adresse im U880-Softwaremonitor: 0BE2H)
Entsprehend im Akkumulator eingetragener Laufwerksnummer wird der READY-Test durchgeführt, d.h. es wird das Signal READY des angeforderten Laufwerks abgefragt. Es gilt:

Laufwerk bereit: Zero-Flag ist auf 0 gesetzt.
Laufwerk nicht bereit: Zero-Flag ist auf 1 gesetzt
Completion code 0C2H

FLOPPY (Adresse im U880-Softwaremonitor: 0BFDH)
Die Floppy-Disk-Routine realisiert das Lesen/schreiben einer bestimmten entsprechend IY-Vektor festgelegten Anzahl von Daten von/auf die Diskette und das Formatieren einer Spur. Der Diskettentyp wird dabei, wie auch beim Lesen und Schreiben, durch den Inhalt von FDCONF (siehe UDOS Systemkommando SETFD) bestimmt. Folgende Requests existieren:

READ BINARY (0AH)
Entsprecehdn IY-Vektor wird eine Anzahl von Daten von der Diskette gelesen und in den Speicher eingetragen. Die Routine kehrt nur mit gesetztem Completion Code in das aufrufende Programm zurück.

READ BINARY - zeitgeteilt (0BH)
Entsprechend IY-Vektor wird eine Anzahl von Daten im zeitgeteilten Betrieb von der Diskette gelesen und in den Speicher eingetragen. Infolge der Interruptstruktur von FLOPPY geht die Routine bei Wartezeiten in das übergeordnete Programm zurück und arbeitet dort weiter. Achtung: Der IY-Vektor darf vor Eintragen des Completion Codes nicht verändert werden.

WRITE BINARY (0EH)
Die Daten werden aus einem bestimmten Speicherbereich geholt und entsprechend IY-Vektor auf die Diskette geschrieben. Die Floppy-routine wird erst mit gesetztem Completion Code wieder verlassen.

WRITE BINARY - zeitgeteilt (0FH)
Die Daten werden im zeitgeteilten Betrieb aus einem bestimmten Speicherbereich geholt und entsprechend IY-Vektor auf die Diskette geschrieben. Durch die Interruptstruktur geht FLOPPY bei Wartezeiten in das aufrufende Programm zurück und arbeitet dort weiter. vor Eintragen des Completion-Codes darf der IY-Vektor nicht verändert werden.

FORMAT (50H)
Die im IY-Vektor angegebene Spur wird entsprechend in FDCONF eingetragenem Format formatiert. Die im IY-Vektor anzugebende Datenlänge ergibt sich aus der Anzahl der Sektoren einer Spur multipliziert mit vier. Die Datenadresse gibt den Anfang des Datenbereichs an, in dem die Sektorinformationen vom Anwender bereitgestellt werden. Jeweils vier Informationen müssen je Sektor zur Verfuegung stehen: Spurnummer, Kopfnummer, Sektornummer, Kennzeichen für Anzahl der Bytes je Sektor. Die Floppy-Routine wird erst mit gesetztem Completion Code wieder verlassen.

Anhang C: U8000-Monitorbeschreibung P8000-Grundgerät

Der U8000-Softwaremonitor setzt Softwareunterbrechungspunkte für die Programmierung.
Ein Unterbrechungspunkt ist ein Befehl, der die Programmabarbeitung an einer festgelegten Adresse unterbricht. Beim Erreichen eines Unterbrechungspunktes werden alle Register, der Stand des Programmzählers (PC) und der Stand des Flag- und Control-Wortes (FCW) gerettet. Die Adresse der Programmunterbrechung wird angezeigt (BREAK AT xxxx).

Eine beliebige Anzahl von Unterbrechungspunkten kann manuell gesetzt werden, indem auf der Unterbrechungsadresse der Befehlskode %7F00 anstelle des Programmbefehlskodes eingetragen wird. Der Unterbrechungspunkt muß immer auf einer geraden Adresse liegen.
Wenn der Unterbrechungspunkt nicht mehr benötigt wird, muß der ursprüngliche Befehlskode wieder eingetragen werden.

Mit dem Kommando "B" (Break) kann ein Unterbrechungspunkt über den U8000-Softwaremonitor automatisch gesetzt werden. Die Break-Routine merkt sich die Unterbrechungsadresse und den ursprünglichen Befehlskode auf dieser Adresse. Wenn der Unterbrechungspunkt gelöscht wird (durch Kommando "B" ohne Parameter oder durch Eingabe eines neuen Unterbrechungspunktes mittels Kommando "B"), so wird der ursprüngliche Befehlskode wieder automatisch eingetragen.
Beim Break-Kommando gibt es die Möglichkeit, einen Schleifenzähler 'n' zu setzen. Das Programm wird dann erst unterbrochen, wenn der Unterbrechungspunkt zum n-ten Mal erreicht wird.
Soll die Programmabarbeitung an einem Unterbrechungspunkt fortgesetzt werden, so kann dies mit den Kommandos "Go" oder "Next" erfolgen, ohne daß dazu der Unterbrechungspunkt gelöscht werden muß.

Folgende Restriktionen für ein Anwenderprogramm existieren für das Setzen von Unterbrechungspunkten, da das Break- und Next-Kommando eine Befehlsmodifikation ausführen und das Interruptsystem benutzen:
- Das zu testende Programm darf nicht im PROM stehen und nicht den Interruptstatus ändern.
- Das zu testende Programm muß in der Lage sein, Interrupts nach dem Erreichen eines Unterbrechungspunktes zu akzeptieren.
- Das Programm sollte nicht zeitabhängig sein, da eine Zeitverzerrung beim Erreichen eines Unterbrechungspunktes eintritt.
- Das Programm darf nicht den Kanal 3 des CTC0 benutzen.
- Der Unterbrechungspunkt darf nicht innerhalb einer Interruptserviceroutine stehen, die durch einen Interrupt der Kanäle 0,1,2 des CTC0 aufgerufen wurde.

Kommandoübersicht

Die folgenden Vereinbarungen werden bei der Kommandobeschreibung verwendet:

() Kennzeichnung von optionalen Parametern
/ logisches ODER

- alle Kommandos und Parameter müssen in Großbuchstaben geschrieben werden
- Zahlen müssen hexadezimal angegeben werden und mit einer Ziffer 0-9 beginnen - ein Hexadezimalkennzeichen wird nicht eingegeben
- fehlerhafte Eingaben (falsche Kommandos, falsche Parameter u.ä.) werden mit dem Fehlerkennzeichen "?" quittiert
- Adressen können eine optionale Segmentangabe beinhalten die bei der Eingabe in spitze Klammern einzuschließen ist

D	Display	adr (datenanzahl) (datentyp_W/B/L)
C	Compare	anfangsadr1 anfangsadr2 byteanzahl
F	Fill	anfangsadr endadr datenwort
M	Move	quelladr zieladr byteanzahl
PR	Port Read	portadr (datentyp_W)
PRS	Port Read Special	portadr (datentyp_W)
PW	Port Write	portadr (datentyp_W) datenwerte
PWS	Port Write Special	portadr (datentyp_W) datenwerte
R	Register	(registername)
B	Break	(adr (schleifenzaehler))
G	Go	(startadr)
N	Next	(befehlsanzahl)
HR	Hard disk Read	blocknr pufferadr (devicenr)
HW	Hard Disk Write	blocknr pufferadr (devicenr)
GE	Get	dateiname (segmentnr)
S	Save	dateiname andangsadr endadr (E=entry_point)(RL=record_length)
Q	Quit	
QRES	Quit-Reset	
RGE	Remote-Get	dateiname (segmentnr)
RS	Remote-Save	dateiname andangsadr endadr (E=entry_point)(RL=record_length)
RQ	Remote-Quit	
T	Test	
O	Operating System	D/U/F/R

Kommando "D" (Display) - Anzeigen bzw. Verändern von Speicherinhalten:

D adresse (datenanzahl) (datentyp_W/B/L)

adresse: Anfangsadresse des anzuzeigenden bzw. zu verändernden Speicherbereiches
datenanzahl: Anzahl (hexadezimal, %1-%FFFF) der anzuzeigenden Datenwerte (BYTE, WORD, LONG):
die Datenanzahl darf nicht mit "B" beginnen ("0B" statt "B" schreiben)
datentyp: Typ der anzuzeigenden bzw. zu verändernden Datenwerte (implizit "W")
W - WORD (16 Bit)
B - BYTE (8 Bit)
L - LONG (32 Bit)

Fehlt die Datenanzahl, wird nur ein Datenwert angezeigt, und es besteht die Möglichkeit, den Speicherinhalt zu verändern.
Beim Verändern von Speicherinhalten können folgende Eingaben erfolgen:

datenwert: Datenwert wird in Speicher geschrieben und Übergang zu nachfolgender Speicheradresse
"_" : Übergang zu vorangehender Speicheradresse
CR : Übergang zu nachfolgender Speicheradresse
"Q" : Verlassen der Display-Routine

Kommando "C" (Compare) - Vergleich zweier Speicherbereiche:

C anfangsadresse1 anfangsadresse2 byteanzahl

anfangsadresse1: Anfangsadresse des 1. Bereiches
anfangsadresse2: Anfangsadresse des 2. Bereiches
byteanzahl: Anzahl (hexadezimal, %1-%FFFF) der zu vergleichenden Bytes

Unterscheiden sich die Inhalte der beiden Speicherbereiche, so werden die unterschiedlichen Speicherinhalte ausgegeben.

Kommando "F" (Fill) - Füllen eines Speicherbereiches mit einem Datenwort:

F anfangsadresse endadresse datenwort

anfangsadresse: Anfangsadresse des Speicherbereiches (muß gerade sein)
endadresse: Endadresse des Speicherbereiches (ohne Segmentnummer - es wird die Segmentnummer der Anfangsadresse verwendet)
datenwort: Datenwort (hexadezimal, %1-%FFFF), mit dem der Speicher gefüllt werden soll

---

Kommando "M" (Move) - Verschieben eines Speicherbereiches:

M quellanfangsadresse zielanfangsadresse byteanzahl

quellanfangsadresse: Anfangsadresse des zu verschiebenden Speicherbereiches
zielanfangsadresse: Anfangsadresse des Zielbereiches
byteanzahl: Anzahl (hexadezimal, %1-%FFFF) der zu verschiebenden Bytes; dabei wird bei 0 der Wert %10000 (64k) angenommen

---

Kommando "PR"/"PRS" (Port Read / Port Read Special) - Lesen eines I/O-Ports bzw. Special-I/O-Ports (MMU):

PR/PRS portadresse (datentyp_W)

portadresse: Adresse des I/O-Ports bzw. Special-I/O-Ports
datentyp: Typ des zu lesenden Datenwertes ("W" - WORD, implizit BYTE)

Kommando "PW"/"PWS" (Port Write / Port Write Special) - Schreiben an einen I/O-Port bzw. Special-I/O-Port (MMU): Port:

PW/PWS portadresse (datentyp_W) datenwerte

portadresse: Adresse des I/O-Ports bzw. Special-I/O-Ports
datentyp: Typ der zu schreibenden Datenwerte ("W" - WORD, implizit BYTE)
datenwerte: ein oder mehrere BYTE- oder WORD-Werte, die an den Port ausgegeben werden sollen

---

Kommando "R" (Register) - Anzeigen bzw. Verändern von Registerinhalten:

R (registerbezeichnung)

registerbezeichnung: eine der folgenden Registerbezeichnung kann angegeben werden:
- (R)0 ... (R)15; (R)SG; (R)PC; FC; (R)RF; (R)N4; (R)N5; (R)PS; (R)P0
- (R)H0; (R)L0 ... (R)H7; (R)L7
- RR0; RR2 ... RR14

Der Inhalt des angegebenen Registers wird angezeigt und es besteht die Möglichkeit, den Registerinhalt zu verändern. Beim Verändern eines Registerinhaltes können folgende Eingaben erfolgen:
datenwert: Register wird mit dem Datenwert beschrieben und Übergang zu nachfolgenden Register
CR : Übergang zu nachfolgenden Register
"Q" : Verlassen der Register-Routine

Der Übergang zu einem nachfolgenden Register erfolgt in der Reihenfolge der oben aufgeführten Registerbezeichnungen einer Zeile.
Fehlt die Angabe einer Registerbezeichnung, so erfolgt die Anzeige aller Registerinhalte.

Kommando "B" (Break) - Setzen bzw. Löschen eines Unterbrechungspunktes:

B (adresse (schleifenzähler))

adresse: Adresse, auf die der Unterbrechungspunkt gesetzt werden soll
(Adresse muß eine RAM-Adresse sein)
schleifenzähler: Anzahl (hexadezimal, %1-%FFFF) der Durchläufe des Unterbrechungspunktes bis eine Programmunterbrechung erfolgen soll (implizit=1)
Wurde nur das Kommando "B" ohne Parameter angegeben, so erfolgt ein Löschen des zuletzt eingegebenen Unterbrechungspunktes.

Kommando "G" (Go) - Programmabarbeitung starten bzw. fortsetzen:

G (startadresse)

startadresse: Adresse, bei der die Programmabarbeitung gestartet werden soll
Wurde keine Startadresse angegeben, so wird die Programmabarbeitung beim aktuellen PC-Stand fortgesetzt.

Kommando "N" (Next) - Schrittweise Programmabarbeitung:

N (befehlsanzahl)

befehlsanzahl: Anzahl (hexadezimal, %1-%FFFF) der im Einzelschrittbetrieb abzuarbeitenden Befehle (implizit 1)
Nach Abarbeitung eines jeden Befehls werden alle Registerinhalte ausgegeben.

Kommando "HR"/"HW" (Hard Disk Read / Hard Disk Write) - Lesen eines Blockes von Hard Disk bzw. Beschreiben eines Blockes auf Hard Disk:

HR/HW blocknummer pufferadresse (devicenummer)

blocknummer: Nummer des zu lesenden Blockes (hexadezimal, %0-%FFFFFF)

pufferadresse: Speicheradresse, ab der der gelesene Blockinhalt (512 Byte) abgespeichert werden soll bzw. Anfangsadresse des Speicherbereiches, mit dessen Inhalt der Block beschrieben werden soll
devicenummer: Gerätenummer 0-3 (implizit 0)

Kommando "GE" (Get) - Laden einer UDOS-Maschinenkodedatei (Typ Procedure) von Diskette (über das UDOS-Betriebssystem des 8-Bit-Mikrorechnerteils) in den RAM des 16-Bit-Mikrorechnerteils des P8000:

GE dateiname (segmentnummer)

dateiname: Name der zu ladenden Datei (UDOS-Dateiname)
segmentnummer: Zielsegment für die zu ladende Datei (implizit 0)
Das Maschinenkodeprogramm wird innerhalb des angegebenen Segments ab der Adresse (Offsetadresse) geladen, wie sie im Deskriptor der UDOS-Datei festgelegt ist. Die UDOS-Datei (Typ Procedure) darf dabei nur aus einem Dateisegment bestehen.
Das Programm wird nicht geladen, wenn sich die Anfangsadresse im Bereich %0000-%7FFF des Segments 0 befindet.

Kommando "S" (Save) - Auslagern eines Speicherausschnittes des 16-Bit-Mikrorechnerteils des P8000 als UDOS-Maschinenkodedatei (Typ Procedure) auf Diskette (über das UDOS-Betriebssystem des 8-Bit-Mikrorechnerteils):

S dateiname anfangsadresse endadresse
(E=entry_point)(RL=record_length)

dateiname: Name der auszulagernden Datei (UDOS-Dateiname)
anfangsadresse: Anfangsadresse des auszulagernden Speicherbereiches
endadresse: Endadresse des auszulagernden Speicherbereiches (ohne Segmentnummer - es wird die Segmentnummer der Anfangsadresse verwendet)
entry_point: Eintrittspunkt, der in den Deskriptor der UDOS-Datei eingetragen werden soll (implizit 0)
record_length: Record-Länge der UDOS-Datei (implizit %100)

Hinweis: Die Kommandos "GE" und "S" können nur benutzt werden, wenn auf dem 8-Bit-Mikrorechnerteil des P8000 das Betriebssystem UDOS läuft, d.h., wenn der U8000-Softwaremonitor mit der WEGA-Startdiskette gestartet wurde.

Kommando "Q" (Quit) - Rückkehr in den 8-Bit-Mikrorechnerteil ohne Rücksetzen des 16-Bit-Mikrorechnerteils (Rückkehr in das System, von dem aus der U8000-Softwaremonitor gestartet wurde (Betriebssystem UDOS bei Start mit WEGA-Startdiskette bzw. U880-Softwaremonitor):

```
Q
```

Kommando "QRES" (Quit-Reset) - Rückkehr in den 8-Bit-Mikrorechnerteil mit Rücksetzen (RESET) des 16-Bit-Mikrorechnerteils (Rückkehr in das System, von dem aus der U8000-Softwaremonitor gestartet wurde (Betriebssystem UDOS bei Start mit WEGA-Startdiskette bzw. U880-Softwaremonitor):

```
QRES
```

Kommando "RGE" (Remote-Get) - Laden einer UDOS-Maschinenkodedatei (Typ Procedure) von einem lokalen System (über ein "remote"-Programm) in den RAM des 16-Bit-Mikrorechnerteils des P8000:

```
RGE dateiname (segmentnummer)
```

dateiname: Name der zu ladenden Datei (UDOS-Dateiname)
segmentnummer: Zielsegment für die zu ladende Datei (implizit 0)

Das Maschinenkodeprogramm wird innerhalb des angegebenen Segments ab der Adresse (Offsetadresse) geladen, wie sie im Deskriptor der UDOS-Datei festgelegt ist. die UDOS-Datei (Typ Procedure) darf dabei nur aus einem Dateisegment bestehen.
Das Programm wird nicht geladen, wen sich die Anfangsadresse im Bereich %0000-%7FFF des Segments 0 befindet.

Kommando "RS" (Remote-Save) - Auslagern eines Speicherausschnittes des 16-Bit-Mikrorechnerteils des P8000 als UDOS-Maschinenkodedatei (Typ Procedure) auf ein lokales System (über ein "remote"-Programm):

```
RS dateiname anfangsadresse endadresse
                    (E=entry_point)(RL=record_length)
```

dateiname: Name der auszulagernden Datei (UDOS-Dateiname)
anfangsadresse: Anfangsadresse des auszulagernden Speicherbereiches
endadresse: Endadresse des auszulagernden Speicherbereiches (ohne Segmentnummer - es wird die

Segmentnummer der Anfangsadresse verwendet)
entry_point: Eintrittspunkt, der in den Deskriptor der UDOS-Datei eingetragen werden soll (implizit 0)
record_length: Record-Länge der UDOS-Datei (implizit %100)

Kommando "RQ" (Remote-Quit) - Rückkehr in das Betriebssystem des lokalen Systems (über ein "remote"-Programm) ohne Beeinflussung des P8000 ("local"-Mode)

RQ

Hinweis: Die Kommandos "RGE", "RS" und "RQ" können nur benutzt werden, wenn anstelle des Terminals für den U8000-Softwaremonitor ein lokales System (z.B. ein Bürocomputer mit dem Betriebssystem UDOS) seriell (V24) angeschlossen wird. Auf dem lokalen System muß dann ein entsprechendes "remote"-Programm laufen, das die V24-Schnittstelle bedient.

Kommando "T" (Test) - Hardwareeigentest des 16-Bit-Mikrorechnerteils des P8000:

T

(Beschreibung siehe Abschn. 3.1.1.)

Kommando "O" (Operating System) - Manueller Start des WEGA-Betriebssystems:

O D/U/F/R

D: Laden und Starten des BOOT-Programms von Hard-Disk (Block 0) mit Kennung für einen manuellen Start
U: Laden und Starten des BOOT-Programms (Datei "boot0.ud") von einer UDOS-Diskette (über das UDOS-Betriebssystem des 8-Bit-Mikrorechnerteils des P8000 - nur aufrufbar, wenn auf dem 8-Bit-Mikrorechnerteils das Betriebssystem UDOS läuft, d.h., wenn der U8000-Softwaremonitor mit der WEGA-Startdiskette gestartet wurde)
F: Laden und Start des BOOT-Programms von einer Diskette (Block 0 einer blockorientierten Diskette im WEGA-Format im Laufwerk 0 - nur aufrufbar, wenn auf dem 8-Bit-Mikrorechnerteil das Betriebssystem UDOS läuft, d.h., wenn der U8000-Softwaremonitor mit der WEGA-Startdiskette gestartet wurde).
R: Lader und Starten des BOOT-Programms (Datei "boot0.rm") von einem lokalen System (über ein entsprechendes "remote"-Programm) - das lokale System ist anstelle des Terminals für den U8000-Softwaremonitor an das P8000-Grundgerät anzuschließen

---

Taste "NMI" (direkt nach Monitormeldung) - Automatischer Start des WEGA-Betriebssystems von Hard-disk:

- Durchführung des Hardwareeigentests des 16-Bit-Mikrorechnerteils (siehe Abschn. 3.1.1.) - bei Fehler erfolgt Sprung in den U8000-Softwaremonitor
- Laden und Starten des BOOT-Programms von Hard-Disk (Block 0) mit Kennung für einen automatischen Start

---

Taste "NMI" (sonst) - Abarbeitung der NMI-Routine:

- Ausgabe Text "NMI"
- Sprung zur Kommandoeingabe

Systemparameter U8000-Softwaremonitor

Die folgenden Systemparameter des U8000-Softwaremonitors sind für den Nutzer zugänglich:

NULLCT: Null Count (%43F6) - initialisiert auf %00
(Anzahl der Nullen, die nach einem "Line Feed" auf das Terminal ausgegeben werden)
LINDEL: Line Delete (%43F3) - initialisiert auf %7F (RUB)
(Zeichen, das das Löschen einer Eingabezeile veranlaßt)
CHRDEL: Character Delete (%43F2) - initialisiert auf %08 (BS, CONTROL-h) - (Zeichen, das das Löschen eines eingegebenen Zeichens veranlaßt)
XOFCHR: XOFF Character (%43F5) - initialisiert auf %13 (CONTROL-s) - (Zeichen, das die Ausgabe auf das Terminal stoppt)
XONCHR: XON Character (%43F4) - initialisiert auf %11 (CONTROL-q) - (Zeichen, das die Ausgabe nach Eingabe von XOFCHR wieder freigibt)
STACK: Stack Pointer (%40A0) - (Stackbereich von %40A0-%4000)
PSAREA: Program Status Area (%4400) - (Beginn der "Program Status Area"-Tabelle)

Der U8000-Softwaremonitor initialisiert folgende "Program Status Area"-Tabelle:

Wort	Wert und Erläuterung
	ID FCW PCSEG PCOFF
0- 3	0000 0000 0000 0000
	reserviert
4- 7	0000 4000 8000 #UNISTR_ERR
	nichtimplementierter Befehl (Trap)
8- B	0000 4000 8000 #PINSTR_ERR
	privilegierter Befehl in Normal-Mode (Trap)
C- F	0000 C000 8000 #SC_ENTRY
	System Call Befehl (Trap)
10-13	0000 4000 8000 #MMU_ERR
	Segment Trap
14-17	0000 4000 8000 #NMI_INT
	Nichtmaskierter Interrupt (NMI)
18-1B	0000 4000 8000 #NVI_ERR
	nichtvektorisierter Interrupt (NVI)
	ID FCW
1C-1D	0000 4000 vektorisierter Interrupt (VI)
	PCSEG PCOFF
1E-1F	8000 #VI_ERR Vektor 0 - unbenutzt
20-21	8000 #VI_ERR Vektor 2 - unbenutzt
22-23	8000 #VI_ERR Vektor 4 - unbenutzt
24-25	8000 #GO_INT/NXT_INT Vektor 6 - CTC0, Kanal 3 (Go, Next)
26-27	8000 #KOPPEL_INT Vektor 8 - PIO1, Kanal B (Zeichen empfangen - Koppelschnittstelle)
28-29	8000 #VI_ERR Vektor A - unbenutzt
2A-2B	8000 #VI_ERR Vektor C - unbenutzt
2C-2D	8000 #VI_ERR Vektor E - unbenutzt
2E-2F	8000 #VI_ERR Vektor 10 - unbenutzt
30-31	8000 #VI_ERR Vektor 12 - unbenutzt
32-33	8000 #PTY_INT Vektor 14 - SIO0, Kanal B (Zeichen empfangen-Terminal)
34-35	8000 #PTY_ERR Vektor 16 - SIO0, Kanal B (Empfangssonderfall-Terminal)
36-37	8000 #VI_ERR Vektor 18 - unbenutzt
38-39	8000 #VI_ERR Vektor 1A - unbenutzt
3A-3B	8000 #LD_INT Vektor 1C - SIO0, Kanal A (Zeichen empfangen - LOAD)
3C-3D	8000 #LD_ERR Vektor 1E - SIO0, Kanal A (Empfangssonderfall - LOAD)
3E-3F	8000 #VI_ERR Vektor 20 - unbenutzt
.	. . . . .
.	. . . . .
.	. . . . .
11C-11D	8000 #VI_ERR Vektor FE - unbenutzt

U8000-Softwaremonitor I/O-Prozeduren

Die Monitorprozeduren, die Ein-/Ausgabefunktionen bezogen auf das P8000-Systemterminals realisieren, können auch durch Systemaufrufe an den U8000-Softwaremonitor in Anwenderprogrammen genutzt werden:

*** TYRD *** Liest ein Zeichen aus dem Terminal-Interrupteingabepuffer. Falls der Puffer leer ist, wird auf die Eingabe eines Zeichens gewartet.

```
Ausgabewerte:       RL3 - gelesenes Zeichen
zerstörte Register: R3
Beispiel:           CONSTANT
                      TYRD := %04
                      ...
                      SC #TYRD
                      (Zeichen in RL3)
```

*** TYWR *** Gibt ein Zeichen auf dem Terminal aus. Das Zeichen wird nicht ausgegeben, falls das XOFF-Zeichen vor Ausführung dieser Prozedur empfangen wurde. In diesem Fall wartet die Prozedur, bis das XON-Zeichen vom Terminal empfangen wird und gibt dann das Zeichen auf dem Terminal aus.

```
Eingabewerte        RL0 - auszugebendes Zeichen
Ausgabewerte:       Z=1, wenn das auszugebende Zeichen ein
                         CR war
zerstörte Register: R3
Beispiel:           CONSTANT
                      TYWR := %06
                      ...
                      (RL0 enthält auszugebendes Zeichen)
                      SC #TYWR
```

*** WR_MSG *** Gibt eine Zeichenkette auf dem Terminal aus (max. 126 Zeichen). Am Anfang der Zeichenkette muss dabei die Anzahl der Zeichen der Zeichenkette stehen (WORD). Enthält die Zeichenkette ein CR, so wird die Zeichenkette nur bis zum Auftreten des ersten CR (einschließlich) ausgegeben.

```
Eingabewerte:       R2 - Anfangsadresse der Zeichenkette
zerstörte Register: R1, R2, R3
Beispiel:           CONSTANT
                      WR_MSG := %0C
                      ...
                      (R2 zeigt auf den Anfang der auszuge-
                      benden Zeichenkette, die in den ers-
                      ten 2 Byte die Länge der Zeichenkette
                      enthält) SC #WR_MSG
```

*** RD_LINE_BFF *** Empfängt eine Zeichenkette vom Terminal bis zum ersten CR und gibt sie auf dem Terminal aus. Die Zeichenkette wird in einem vom Nutzer bereitgestellten Puffer abgespeichert. Kleinbuchstaben werden vorher in Großbuchstaben umgewandelt.

```
Eingabewerte:       R2 - Adresse des Eingabepuffers
                    R1 - Länge des Eingabepuffers
Ausgabewerte:       R1 - Länge der empfangenen Zeichenkette
                    Z=1, wenn Eingabepuffer voll ist
zerstörte Register: R2, R3
Beispiel:           CONSTANT
                      RD_LINE_BFF := %08
                      ...
                      (Adresse des Eingabepuffers in R2,
                      Pufferlänge in R1)
                      SC #RD_LINE_BFF
                      (Länge der Zeichenkette in R1)
```

*** WR_CRLF *** Ausgabe von CR und LF auf Terminal

```
zerstörte Register: R3
Beispiel:           CONSTANT
                      WR_CRLF := %0A
                      ,,,
                      SC #WR_CRLF
```

Kombinat VEB
ELEKTRO-APPARATE-WERKE
BERLIN-TREPTOW >FRIEDRICH EBERT<
Hoffmannstraße 15-26, Berlin, DDR-1193
011 2263 eaw 011 2264 eaw

Die Angaben über technische Daten entsprechen dem bei Redaktionsschluß vorliegenden Stand. Änderungen im Sinne der technischen Weiterentwicklung behalten wir uns vor.