

---

# **CP32 MINICONTROL ZENTRALEINHEIT**

## **KURZBESCHREIBUNG**

**Version:** 1.0 (Juli 1992)

**Herausgeber:** Bernecker und Rainer Industrie-Elektronik GmbH.

**Best. Nr.:** MACP32KB-0

---

---

# MINICONTROL ZENTRALEINHEIT CP32

---

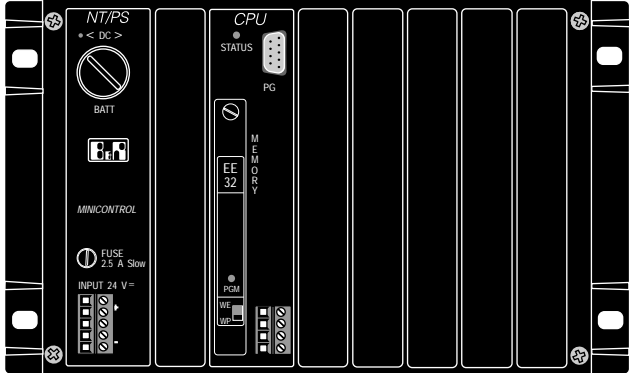
<b>Inhalt:</b>		
	Bestellnummern - Bestellbezeichnungen	4
	Steckplätze	4
	Technische Daten	5
	Online-Schnittstelle	6
	Anwenderschnittstelle	7
	Status-LED	11
	Befehlssatz	12
	Mathematik-Routinen	13
	Speicheraufteilung	16
	System-Speicherstellen	17
	First Scan-Flag	18
	Batteriekontrolle	18
	Zeittakte	19
	Zeitimpulse	19
	Echtzeituhr	20
	Softwarezeiten	21
	Inport/Outport Adresse \$3400	23
	Zusätzliches Anwender-EEPROM	24
	Inport Adresse \$3480	30
	Runtime-Überwachung	31
	Timerinterrupt-Routinen	31
	Fehlermeldungen	32
	Anwenderprogrammspeicher	35
	Internes RAM	36
	EE32 - RAM/EEPROM-Anwenderprogrammspeicher	36
	EP05 - EPROM-Anwenderprogrammspeicher	38
	Einschaltverhalten	39



## BESTELLNUMMERN - BESTELLBEZEICHNUNGEN

Die Zentraleinheit CP32 ist Bestandteil der MINICONTROL Grundeinheit C (Best.Nr. MCGE232-022).

MINICONTROL  
Grundeinheit C

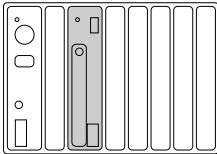


**MINICONTROL Grundeinheit C**, bestehend aus Gehäuse, Zentraleinheit CP32, serielle RS485/TTY-Anwenderschnittstelle, Echtzeituhr, 32 KByte EEPROM-Zusatzspeicher (Daten), Stromversorgungsmodul NT33 und Anwenderprogrammspeichermodul (16 KByte RAM, 16 KByte EEPROM für 4,7 K Anwendungen), 6 Steckplätze für digitale E/A-Module und Zeitmodule, davon 2 geeignet für den Betrieb von analogen E/A-Modulen, Schnittstellenmodulen und Zählmodulen.

**MCGE232-022**

## STECKPLÄTZE

Die MINICONTROL Zentraleinheit CP32 darf nur auf dem grau gekennzeichneten Steckplatz betrieben werden:



**MINICONTROL**

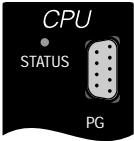
## TECHNISCHE DATEN

	<b>CP32</b>
Prozessor	6303
Bearbeitungszeit	ca. 4 ms / k Anweisungen
Anwenderprogrammspeicher Größe Art Ausführung	16 kByte RAM/EEPROM oder EPROM intern oder von vorne steckbar
EEPROM Erweiterungsspeicher	32 KByte
Status-LED	JA
Anzahl E/A digital analog	192 16
Serielle Schnittstellen Online-Schnittstelle Anwender-Schnittstelle	TTY TTY/RS485 (umschaltbar)
Anzahl 8 Bit-Speicher remanent nicht remanent	7168 7148 20
Anzahl 1 Bit-Speicher remanent nicht remanent	1000 500 500
Uhrzeit/Datum	Echtzeituhr
Hardware-Timer	24
Software-Timer	64
Zeittakte/Zeitimpulse	10 ms, 100 ms, 1 s, 10 s
Betriebstemperatur	0 bis 60 °C
Luftfeuchtigkeit	0 bis 95 %, nicht kondensierend
Batterieüberwachung	JA
Watchdog	JA

# ONLINE-SCHNITTSTELLE

Zur Kommunikation mit dem Programmiergerät verfügt die Zentraleinheit über eine Online-Schnittstelle. Die Online-Schnittstelle ist eine TTY-Schnittstelle mit 62,5 kBaud, die für den Onlinebetrieb mit dem Programmiergerät verwendet werden kann.

Die Online-Schnittstelle ist an der Modulfront mit "PG" gekennzeichnet:



## Pinbelegung der Online-Schnittstelle

	Pin	Funktion
	1	TXD
	2	reserviert
	3	RXD RET
	4	Reset RET
	5	reserviert
	6	TXD RET
	7	RXD
	8	Reset
	9	reserviert

## Online-Kabel


Online-Kabel Best. Nr.	für Interface/PG <sup>1)</sup>
BRKAOL-0	BRIFPC-0 BRIFTO-0 BRADFOL
BRKAOL2-0	BRIFCO-0

<sup>1)</sup> Alle in diesem Handbuch beschriebenen PG-Funktionen beziehen sich auf das B&R-Programmiersystem V 5.0.

## ANWENDER-SCHNITTSTELLE

Die Zentraleinheit CP32 verfügt über eine TTY/RS485-Anwenderschnittstelle (umschaltbar, galvanisch getrennt).

- Hinweis:**
- Die maximale Übertragungsrate der TTY bzw. RS485 Schnittstelle ist 19200 Baud.
  - Alle Schnittstellen sind schutzbeschaltet und besitzen ein Eingangsfilter.

	Pin	TTY	RS485
	1	TXD	
	2	TXD RET	Data
	3	RXD	Data
	4	RXD RET	

### UMSCHALTUNG TTY/RS485

Nach dem Einschalten ist die Schnittstelle auf TTY eingestellt. Die Umschaltung zwischen TTY und RS485 Schnittstelle erfolgt mit Bit 4 der Inport/Outport Adresse \$3400.

```
0 ... TTY
1 ... RS485
```

Wird die Schnittstelle in der Initialisierungsroutine umgeschaltet, ist darauf zu achten, daß die Verzögerung maximal 10 ms beträgt (Relaisverzögerung).

**Beispiel:** Umschalten der Schnittstelle von TTY auf RS485.

```
LD    # $3400          ERD mit $3400 laden
DXR                    Indexregister auf Adresse $3400 setzen
LAD    I 000            Inhalt in ERA laden
OD     # %00010000      Bit 4 auf 1 setzen
=      I 000            ERA in Adresse $3400 speichern
```

### SOFTWAREMÄSSIGE BEDIENUNG

Die softwaremäßige Bedienung der Anwenderschnittstelle erfolgt über die folgenden Register:

```
P 103    Programmregister
P 102    Befehlsregister
P 101    Statusregister
P 100    Datenregister
```

## Initialisierung

Bei der Initialisierung werden Programmregister und Befehlsregister mit bestimmten Vorwahlwerten beschrieben. Dadurch werden Baudrate, Datenformat, Parity usw. festgelegt. Die Initialisierung wird nur ein mal unmittelbar nach dem Einschalten der SPS oder nach einem Reset durchgeführt.

<div><div>70</div><div><div>SB</div><div>DB</div><div>1</div><div>BAUD</div></div></div> <div>P 103</div>	<table><tr><td><b>SB</b></td><td>Anzahl Stopbits</td><td>0 ... 1 ...</td><td>1 Stopbit wenn DB=5 und kein Parity ... 1,5 Stopbits wenn DB=8 und Parity ... 1 Stopbit in allen anderen Fällen ... 2 Stopbits</td></tr><tr><td><b>DB</b></td><td>Anzahl Datenbits</td><td>00 ... 01 ...</td><td>8 Datenbits 7 Datenbits</td><td>10 ... 11 ...</td><td>6 Datenbits 5 Datenbits</td></tr><tr><td><b>BAUD</b></td><td>Baudrate</td><td>0001 ... 0010 ... 0011 ... 0100 ... 0101 ...</td><td>50 75 109,92 134,58 150</td><td>0110 ... 0111 ... 1000 ... 1001 ... 1010 ...</td><td>300 600 1200 1800 2400</td><td>1011 ... 1100 ... 1101 ... 1110 ... 1111 ...</td><td>3600 4800 7200 9600 19200</td></tr></table>	<b>SB</b>	Anzahl Stopbits	0 ... 1 ...	1 Stopbit wenn DB=5 und kein Parity ... 1,5 Stopbits wenn DB=8 und Parity ... 1 Stopbit in allen anderen Fällen ... 2 Stopbits	<b>DB</b>	Anzahl Datenbits	00 ... 01 ...	8 Datenbits 7 Datenbits	10 ... 11 ...	6 Datenbits 5 Datenbits	<b>BAUD</b>	Baudrate	0001 ... 0010 ... 0011 ... 0100 ... 0101 ...	50 75 109,92 134,58 150	0110 ... 0111 ... 1000 ... 1001 ... 1010 ...	300 600 1200 1800 2400	1011 ... 1100 ... 1101 ... 1110 ... 1111 ...	3600 4800 7200 9600 19200
<b>SB</b>	Anzahl Stopbits	0 ... 1 ...	1 Stopbit wenn DB=5 und kein Parity ... 1,5 Stopbits wenn DB=8 und Parity ... 1 Stopbit in allen anderen Fällen ... 2 Stopbits																
<b>DB</b>	Anzahl Datenbits	00 ... 01 ...	8 Datenbits 7 Datenbits	10 ... 11 ...	6 Datenbits 5 Datenbits														
<b>BAUD</b>	Baudrate	0001 ... 0010 ... 0011 ... 0100 ... 0101 ...	50 75 109,92 134,58 150	0110 ... 0111 ... 1000 ... 1001 ... 1010 ...	300 600 1200 1800 2400	1011 ... 1100 ... 1101 ... 1110 ... 1111 ...	3600 4800 7200 9600 19200												
<div><div>70</div><div><div>PAR</div><div>P<sub>on</sub></div><div>E</div><div>RT</div><div>0</div><div>1</div><div>1</div></div></div> <div>P 102</div>	<table><tr><td><b>PAR</b></td><td>Parity</td><td>00 ... 01 ... 10 ... 11 ...</td><td>Parity ungerade (odd) Parity gerade (even) Parity-Bit beim Senden gesetzt Parity-Bit beim Senden gelöscht</td></tr><tr><td><b>P<sub>on</sub></b></td><td>Parity ein/aus</td><td>0 ... 1 ...</td><td>Kein Parity-Test, Parity-Bit wird nicht generiert Parity-Test aktiv</td></tr><tr><td><b>E</b></td><td>Echo-Mode</td><td>0 ... 1 ...</td><td>Echo-Mode aus Echo-Mode ein, RT muß 0 sein</td></tr><tr><td><b>RT</b></td><td>RTS-Leitung<sup>1)</sup></td><td>0 ... 1 ...</td><td>RTS high, nicht sendebereit RTS low, sendebereit</td></tr></table> <div>TTY: RT = 1</div>	<b>PAR</b>	Parity	00 ... 01 ... 10 ... 11 ...	Parity ungerade (odd) Parity gerade (even) Parity-Bit beim Senden gesetzt Parity-Bit beim Senden gelöscht	<b>P<sub>on</sub></b>	Parity ein/aus	0 ... 1 ...	Kein Parity-Test, Parity-Bit wird nicht generiert Parity-Test aktiv	<b>E</b>	Echo-Mode	0 ... 1 ...	Echo-Mode aus Echo-Mode ein, RT muß 0 sein	<b>RT</b>	RTS-Leitung <sup>1)</sup>	0 ... 1 ...	RTS high, nicht sendebereit RTS low, sendebereit		
<b>PAR</b>	Parity	00 ... 01 ... 10 ... 11 ...	Parity ungerade (odd) Parity gerade (even) Parity-Bit beim Senden gesetzt Parity-Bit beim Senden gelöscht																
<b>P<sub>on</sub></b>	Parity ein/aus	0 ... 1 ...	Kein Parity-Test, Parity-Bit wird nicht generiert Parity-Test aktiv																
<b>E</b>	Echo-Mode	0 ... 1 ...	Echo-Mode aus Echo-Mode ein, RT muß 0 sein																
<b>RT</b>	RTS-Leitung <sup>1)</sup>	0 ... 1 ...	RTS high, nicht sendebereit RTS low, sendebereit																

### Beispiel:

Initialisierung der Anwenderschnittstelle, Baudrate = 9600, 8 Datenbits, 1 Stopbit, Parity aus, Echo-Mode aus.

```

LB      # %00011110      9600 Baud, 8 Datenbits, 1 Stopbit
LAD     # %00001011      Parity aus, Echo-Mode aus
=D      P 102            Programmregister & Befehlsregister

```

<sup>1)</sup> Um den RS485 Sender einzuschalten muß das RT Bit auf 1 gesetzt werden. Ab jetzt läuft die Timerzeit ab. Wird kein Zeichen gesendet, schaltet der Sender nach ca. 300 ms wieder ab (hochohmig).

Nach Beschreiben des Datenregisters mit dem zu sendenden Byte muß das RT Bit wieder rückgesetzt werden. Der Bus bleibt bis zur vollständigen Sendung des Zeichens aktiv (die maximale RTS Verzögerung beträgt 5 µs).

Das Umschalten der Handshake-Leitung RTS von low auf high (von 1 auf 0) kann jederzeit erfolgen.

Wenn kein Sender aktiv und der Bus somit hochohmig ist, muß darauf geachtet werden, daß in diesem Zustand undefinierte Zeichen empfangen werden können.





**Zeichen ausgeben** Vor dem Beschreiben des Datenregisters mit dem auszugebenden Zeichen ist zu überprüfen, ob der Sender bereit ist, ein Zeichen zu senden (Bit 4 im Statusregister muß 1 sein).

LB	P 101	Statusregister
BB	# %00010000	Sender bereit ?
SP0	NO	Sprung, wenn Sender nicht bereit
LAD	x xxx	auszugebendes Zeichen
=	P 100	Datenregister

**Zeichen einlesen** Durch Auswerten des Bits 3 im Statusregister wird festgestellt, ob ein Zeichen empfangen wurde. Ist dieses Bit = 1, so wurde ein Zeichen empfangen. Die Bits 0 bis 2 des Statusregisters geben an, ob Übertragungsfehler aufgetreten sind (Parity-Fehler, Overrun-Fehler oder Framing-Fehler). Ist eines dieser Fehlerbits gesetzt, so ist das empfangene Zeichen ungültig. Das Datenregister muß aber auch im Fehlerfall ausgelesen werden, da dadurch die Fehlermeldung quittiert wird.

LB	P 101	Statusregister
BB	# %00001000	Zeichen empfangen ?
SP0	NO	Sprung, wenn kein Zeichen empfangen
LAD	P 100	Datenregister auslesen
BB	# %00000111	Übertragungsfehler aufgetreten ?
SN0	FAIL	Sprung, wenn Übertragungsfehler
:		Auswerten des empfangenen Zeichens

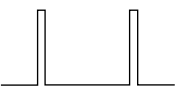
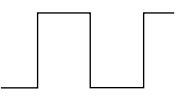
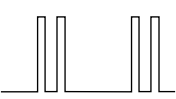
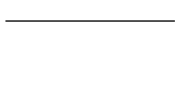

FAIL :

## STATUS-LED

Die MINICONTROL Zentraleinheit CP32 ist mit einer Status-LED ausgestattet, die verschiedene Betriebszustände anzeigt.



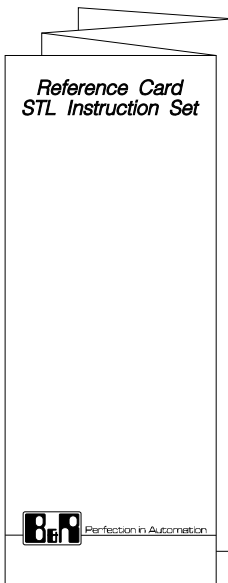
Die folgenden Betriebszustände werden durch unterschiedliche Blinktakte angezeigt:

Blinktakt	Funktion
HI  LO	Anwenderprogramm läuft im RAM
HI  LO	Zentraleinheit ist im HALT-Zustand
HI  LO	Onlinekabel während PROM-Programmieren abgesteckt
HI  LO	Fehler bei der Ausführung des Anwenderprogrammes
HI  LO	Anwenderprogramm läuft im PROM

## BEFEHLSSATZ

In den MINICONTROL Zentraleinheiten wird ein 6303-Prozessor (Hitachi) verwendet. Das ist der selbe Prozessor, der auch in den Zentraleinheiten CP40 (MULTICONTROL), CP41 (MIDICONTROL) und in den PP40 Peripherieprozessoren (MULTICONTROL, MIDICONTROL) zur Anwendung kommt. Dadurch ist volle Software-Kompatibilität zu den anderen SPS-Systemen gegeben.

Eine vollständige Beschreibung des Befehlssatzes des 6303-Prozessors ist in der Kurzbeschreibung "AWL Befehlsbeschreibung" (Best. Nr. MAAWLKB-D - lieferbar Ende 1991) zu finden. In der Faltkarte "STL Instruction Set" (Best. Nr. MASTL-E) sind alle Befehle tabellarisch zusammengefaßt.



Diese Faltkarte enthält u.A. folgende Informationen:

- B&R- und MOTOROLA-Mnemonics
- Befehlsbeschreibung
- Mögliche Adressierungsarten
- Mögliche Adreßvorwahlen
- Länge und Dauer der Befehle
- Veränderte Flags

## MATHEMATIK-ROUTINEN

Die MINICONTROL-Zentraleinheiten sind standardmäßig mit schnellen Fließkomma Mathematik-Routinen ausgestattet. Diese Routinen sind Bestandteil des Betriebssystems. Sie werden durch Befehls-Mnemonics aus der Anweisungsliste aufgerufen. Neben den Grundrechenarten Addition, Subtraktion, Multiplikation, Division und Quadratwurzel stehen zahlreiche Umwandlungs- und Hilfsprogramme zur Verfügung (z.B. zum Vergleichen oder Kopieren). Zur Zahlendarstellung wird das genormte 4 Byte IEEE-Format verwendet. Eine detaillierte Beschreibung der Mathematik-Routinen ist in der Kurzbeschreibung MAAWLKB-D (lieferbar Ende 1991) zu finden.

**ACHTUNG** MATHEMATIK-ROUTINEN DÜRFEN NICHT IN INTERRUPTPROGRAMMEN VERWENDET WERDEN.

## ZAHLENFORMATE

Format	Zahlenbereich
<b>IEEE-Fließkommaformat</b> <div><div><div>3124231615870</div><div><div>S</div><div>EXP</div><div>S</div><div>MANTISSE</div></div></div><div>S ... VorzeichenEXP ... 7 Bit ExponentMANTISSE ... 23 Bit Mantisse</div></div>	$-9,22 \cdot 10^{18}$ bis $-9,22 \cdot 10^{-18}$ und $9,22 \cdot 10^{-18}$ bis $9,22 \cdot 10^{18}$
<b>Absolut mit Vorzeichen lang</b> <div><div><div>3124231615870</div><div><div>S</div><div>ABSOLUTBETRAG</div></div></div></div>	$\pm 2,15 \cdot 10^9$
<b>Absolut mit Vorzeichen kurz</b> <div><div><div>1514870</div><div><div>S</div><div>ABSOLUTBETRAG</div></div></div></div>	$\pm 32767$
<b>Integer lang</b> <div><div><div>3124231615870</div><div><div>S</div><div>2er-Komplement</div></div></div></div>	$\pm 2,15 \cdot 10^9$
<b>Integer kurz</b> <div><div><div>1514870</div><div><div>S</div><div>2er-Komplement</div></div></div></div>	$-32768$ bis $+32767$

Bef.	Funktion	Quelle bzw. Operanden	Ziel bzw. Ergebnis	Ausführungszeit in µs	Mögliche Fehlermeldungen													
					1	2	3	4	5	6	7	8	9	10	11	12	13	14
MADD	OP1 := OP1 + OP2	OP1, OP2	OP1	209/690	●	●				●	●					●		
MSUB	OP1 := OP1 - OP2	OP1, OP2	OP1	219/700	●	●				●	●					●		
MMUL	OP1 := OP1 * OP2	OP1, OP2	OP1	209/803	●	●				●	●					●		
MDIV	OP1 := OP1 / OP2	OP1, OP2	OP1	190/1980	●	●	●			●	●					●		
MSQR	OP1 := SQR(OP1)	OP1	OP1	71/8065	●	●				●	●	●				●		
MSGN	OP1 := OP1 * (-1)	OP1	OP1	85/85														
MCOP	OP2 := OP1	OP1	OP2	46/46														
MEXG	OP1 ↔ OP2	OP1, OP2	OP2, OP1	76/76														
LAL1	Lade OP1, abs. mit Vz. 4 Byte	(R)	OP1	190/339						●								
LAL2	Lade OP2, abs. mit Vz. 4 Byte	(R)	OP2	190/339						●								
LAW1	Lade OP1, abs. mit Vz. 2 Byte	ERD	OP1	83/250														
LAW2	Lade OP2, abs. mit Vz. 2 Byte	ERD	OP2	83/250														
LIL1	Lade OP1, int. 4 Byte	(R)	OP1	197/381						●								
LIL2	Lade OP2, int. 4 Byte	(R)	OP2	194/378						●								
LIW1	Lade OP1, int. 2 Byte	ERD	OP1	87/260														
LIW2	Lade OP2, int. 2 Byte	ERD	OP2	84/257														
LF1	Lade OP1, IEEE	(R)	OP1	88/125	●					●	●					●		
LF2	Lade OP2, IEEE	(R)	OP2	88/125	●					●	●					●		
CAF	ASCII - IEEE	(R)	OP1	280/2140	●					●	●		●					
SAW	Speichere OP1, abs. mit Vz. 2 Byte	OP1	ERD	158/373				●								●		
SAL	Speichere OP1, abs. mit Vz. 4 Byte	OP1	(R)	169/408				●								●		
SIW	Speichere OP1, int. 2 Byte	OP1	ERD	158/380				●								●		
SIL	Speichere OP1, int. 4 Byte	OP1	(R)	172/424				●								●		
SFX	Speichere OP1, IEEE	OP1	(R)	43/43														
CFA	OP1 - ASCII	OP1	(R)	352/7310	●			●		●	●					●		
CFa0	OP1 - ASCII mit Vornullen	OP1	(R)	310/7190	●			●		●	●					●		
CFEA	OP1 - ASCII mit Exp.	OP1	(R)	570/7140	●					●	●					●		
SFM1	Speichere OP1 in Speicher 1	OP1	MEM1	60/60														
SFM2	Speichere OP1 in Speicher 2	OP1	MEM2	60/60														
SFM3	Speichere OP1 in Speicher 3	OP1	MEM3	60/60														
RFM1	Lade OP2 aus Speicher 1	MEM1	OP2	56/56														
RFM2	Lade OP2 aus Speicher 2	MEM2	OP2	56/56														
RFM3	Lade OP2 aus Speicher 3	MEM3	OP2	56/56														
FM2B	Multiplikation 2 x 2 Byte	(R) int. 2 Byte, ERD	(C1048, 1049) 4 Byte	115/191														
FM3B	Multiplikation 3 x 2 Byte	(R) int. 3 Byte, ERD	(C1048, 1049) 5 Byte	156/270														
FM4B	Multiplikation 4 x 2 Byte	(R) int. 4 Byte, ERD	(C1048, 1049) 6 Byte	192/344														
CB CD	Binär - BCD	(ERD) abs. 3 Byte	(R) BCD 3 Byte	192/1180				●										
CB IN	BCD - Binär	(ERD) BCD 3 Byte	(R) abs. 3 Byte	112/223														
CIA	Binär - ASCII	(C1048, 1049)	(R)	380/2020				●										
CIA0	Binär - ASCII mit Vornullen	(C1048, 1049)	(R)	310/1960				●										
CBPP	Binär - physikalisch (Parameterber.)	(R)	(C1048, 1049)	2500/6700	●					●	●					●		
CBPQ	Binär - physikalisch (schnell)	ERD, (R)	ERD, OP1	780/1700	●					●	●					●		
CPBQ	Physikalisch - binär (schnell)	ERD, (R)	ERD, OP1	780/1500	●					●	●					●		
CBP	Binär - physikalisch	(C1046, 1047), (R)	(C1048, 1049), ERD, OP1	3400/8300	●					●	●					●		
CPB	Physikalisch - binär	(C1046, 1047), (R)	(C1048, 1049), ERD, OP1	3400/8300	●					●	●					●		
CIM	Inch - metrisch	(C1046, 1047), ERD	(R), ERD	307/472													●	●
CMI	Metrisch - Inch	(C1046, 1047), ERD	(R), ERD	307/472													●	●
FCOP	Speicherbereich kopieren	(R), ERD	(C1048, 1049)															
FSMB	Speicher mit Byte-Werten laden	(R), ERD, C1052	(R)	48 + L * 12														
FSMW	Speicher mit Wort-Werten laden	(R), ERD, C1052	(R)	48 + L * 14														
FCLR	Speicherbereich löschen	(R), ERD	(R)	48 + L * 12														
MCMP	OP1 mit OP2 vergleichen	OP1, OP2		201/223												●		
MHIL	Wenn OP1 > OP2 dann OP1 := OP2	OP1, OP2	OP1	215/271												●		
MLOL	Wenn OP1 < OP2 dann OP1 := OP2	OP1, OP2	OP1	215/271												●		

## FEHLERMELDUNGEN

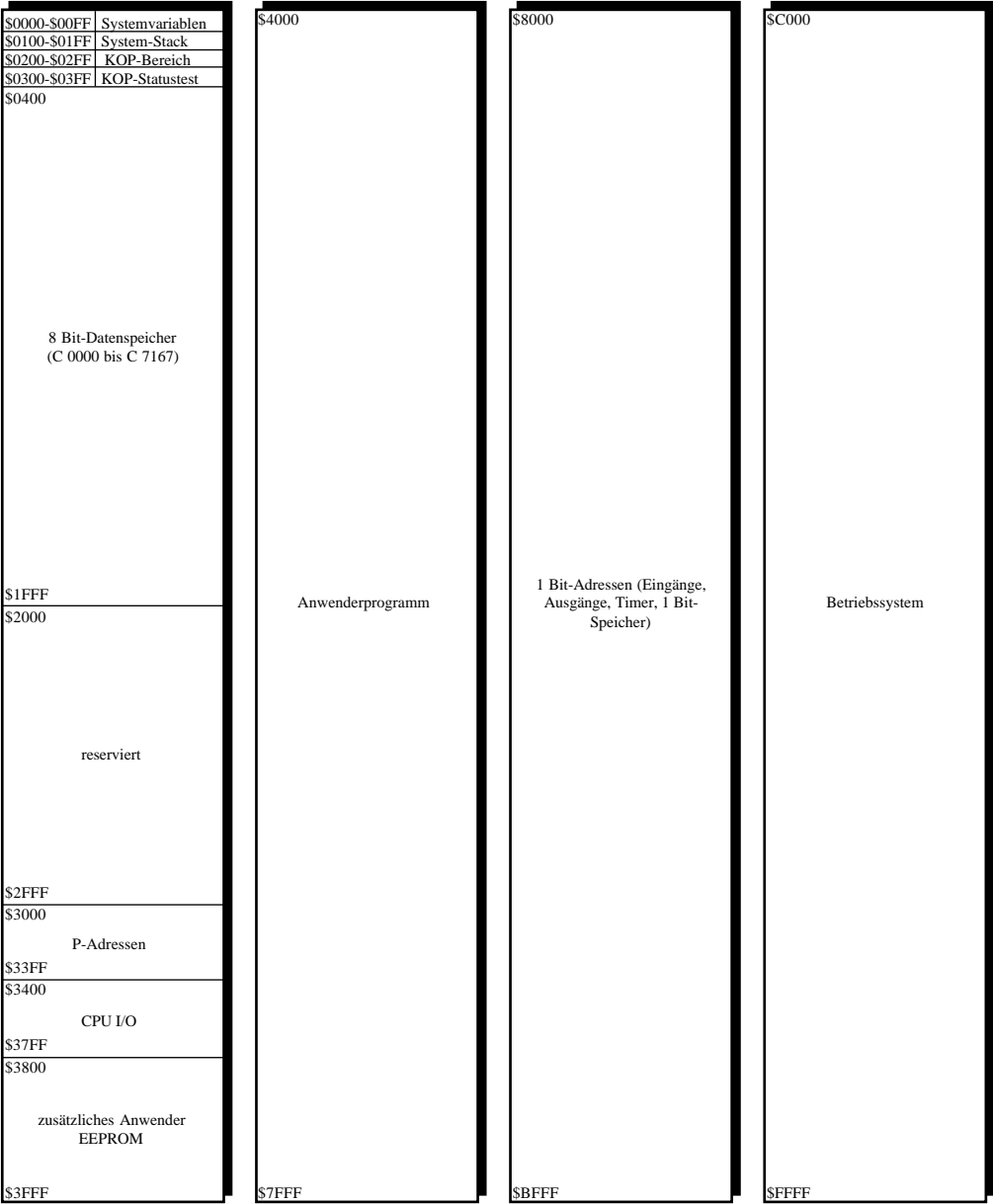
Die in der Tabelle mit ● gekennzeichneten Fehlermeldungen sind für die jeweilige Funktion möglich. Tritt bei der Ausführung einer Routine ein Fehler auf, so wird das Carry-Flag gesetzt und die Speicherstelle C 1024 enthält die Fehlernummer.

Nr	Beschreibung
1	Bei einer Berechnung wurde der darstellbare Zahlenbereich überschritten
2	Bei einer Berechnung wurde der darstellbare Zahlenbereich unterschritten
3	Division durch 0
4	Bereichsüberschreitung beim Umwandeln von Zahlenformaten
5	Beschneidung des Lower Significant Byte (LSB) beim Laden von 4 Byte-Mantissen
6	Bereichsüberschreitung beim Laden von Zahlen
7	Bereichsunterschreitung beim Laden von Zahlen
8	Negativer Operand bei Quadratwurzelberechnung
9	Unzulässiges Zeichen bei Stringumwandlungsroutine
10	nicht verwendet
11	Unzulässiges Kommando (TRAP-Fehler wird ausgelöst)
12	Zahl nicht im Rechenbereich
13	Exponentfehler bei Inch-Metrisch- bzw. Metrisch-Inch-Umwandlung
14	Datenüberlauf bei Inch-Metrisch- bzw. Metrisch-Inch-Umwandlung

## OPERANDEN UND SPEICHER

Speicherstelle(n)	Funktion
C 1024	Fehlernummer
C 1025	reserviert
C 1026 bis C 1029	Operand 1 (OP1)
C 1030 bis C 1033	Operand 2 (OP2)
C 1034 bis C 1037	Zwischenspeicher 1 (MEM1)
C 1038 bis C 1041	Zwischenspeicher 2 (MEM2)
C 1042 bis C 1045	Zwischenspeicher 3 (MEM3)
C 1046 bis C 1047	Quelladresse
C 1048 bis C 1049	Zieladresse
C 1050 bis C 1051	Länge
C 1052 bis C 1053	Daten

SPEICHERAUFTeilUNG





## SYSTEM-SPEICHERSTELLEN

Einige 8 Bit-Speicher und 1 Bit-Speicher sind für Betriebssystemfunktionen reserviert. Diese dürfen vom Anwenderprogramm nicht bzw. nur eingeschränkt verwendet werden:

8 Bit-Speicher:	C 0800 bis C 1499
1 Bit-Speicher:	M 800 bis M 999

1 Bit-Speicher mit Adressen ab M 800, die für Betriebssystem-Sonderfunktionen verwendet sind, werden mit Adressen F Dxx bzw. Z Dxx eingegeben:

Adresse	Einzugeben als <sup>1)</sup>
M 800	F D00
M 801	F D01
:	:
M 899	F D99
M 900	Z D00
M 901	Z D01
:	:
M 999	Z D99

<sup>1)</sup> Das Programmiergerät erlaubt auch die Eingabe der M-Adresse, nach Abschluß der Eingabe mit ENTER wird die Adresse automatisch in die Form F Dxx oder Z Dxx umgewandelt. Z.B.:

Eingabe: M 820  
Wird nach ENTER geändert in: F D20

Eingabe: M 980  
Wird nach ENTER geändert in: Z D80

Im folgenden Abschnitt sind die System-Speicherstellen beschrieben, die vom Anwenderprogramm nur eingeschränkt verwendet werden dürfen:

Zulässiger Zugriff		Adresse(n)	Funktion
Lesen	Schreiben		
✓		C 0800 bis C 0863 C 0899 C 0900 bis C 0963 C 0972, C 0973 C 0974, C 0975 C 0978, C 0979	Vorteiler für Softwarezeiten First Scan-Flag Zähler für Softwarezeiten Timerinterrupt-Vektor Timerinterrupt-Zeit Trap-Vektor
✓	✓	C 0980 bis C 0984	Echtzeituhr
✓	✓	C 0990 C 0991 bis C 0993 C 0998, C 0999	Breakpoint-Sonderfunktion Zähler/Teiler Runtime-Überwachung
✓	✓	C 1024 bis C 1053 C 1054 bis C 1499	Operanden u. Speicher der Mathematik-Routinen Reserviert für Standard-Funktionsbausteine
✓	✓	F D00 bis F D63	Freigaben für Softwarezeiten
✓	✓	F D85, F D86	Steuerbits für Echtzeituhr
✓		Z D00 bis Z D63	Softwarezeiten
✓		Z D64	First Scan-Flag
✓		Z D80 bis Z D83	Zeittakte
✓		Z D90 bis Z D93	Zeitimpulse
✓		Z D99	Batteriekontrolle

## FIRST SCAN-FLAG

Das First Scan-Flag ist eine 1 Bit-Speicherstelle (Z D64), die vom Betriebssystem automatisch während des ersten Programmzyklus auf 1 gesetzt wird, sonst ist dieses Flag 0. Das First Scan-Flag wird für Programminitialisierungen verwendet. Auch die Speicherstelle C 0899 liefert die First Scan-Funktion:

Z D64	First Scan-Flag (1 = erster Programmzyklus)
C 0899	First Scan-Flag (1 = erster Programmzyklus)

### Beispiel:

INIT	LAD	Z D64	First Scan
	SP0	INIR	Sprung, wenn schon initialisiert
	:		
	:		Initialisierungen
	:		
INIR	RET		

Im Funktionsplan kann das First Scan-Flag an den Enable-Eingang von Funktionsbausteinen angeschlossen werden, die nur ein mal während des ersten Programmzyklus ausgeführt werden sollen.

### ACHTUNG:

Mit dem Kommando XFER des B&R Programmiersystemes können Programme ohne Unterbrechung des laufenden Anwenderprogrammes in den RAM-Speicher der Zentraleinheit übertragen werden. Der Anwender muß nach erfolgter Übertragung manuell mit einem Befehl vom Programmiergerät auf das neue Programm umschalten. In diesem Fall sind die First Scan-Speicherstellen während des ersten Programmzyklus des neuen Programmes nicht gesetzt!

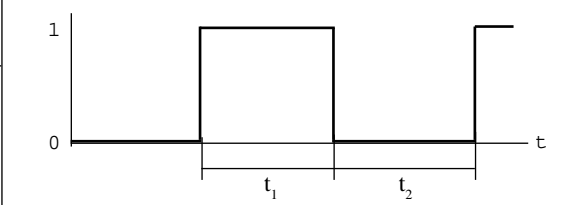
## BATTERIEKONTROLLE

Der Zustand der Batterie wird mit der 1 Bit-Speicherstelle Z D99 kontrolliert.

0	...	Batterie OK	(Spannung > 2,85 V)
1	...	Batterie leer	(Spannung < 2,60 V)

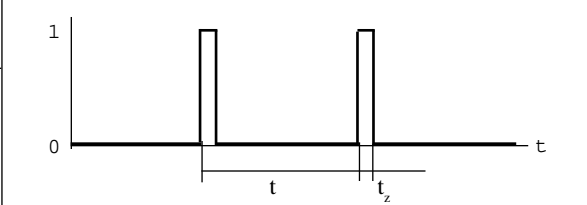
## ZEITAKTE

Zeittakte sind 1 Bit-Adressen, die vom Betriebssystem automatisch mit Blinktakten angesteuert werden:

Adresse	t1	t2	
Z D80	10 ms	10 ms	
Z D81	40 ms	60 ms	
Z D82	0,4 s	0,6 s	
Z D83	4 s	6 s	

## ZEITIMPULSE

Zeitimpulse sind 1 Bit-Adressen, die vom Betriebssystem automatisch für die Dauer eines Programmzyklus auf 1 gesetzt werden.

Adresse	t	
Z D90	10 ms	
Z D91	100 ms	
Z D92	1 s	
Z D93	10 s	

$t_z$  ... Programmzyklus

## ECHTZEITUHR

Wenn die SPS ausgeschaltet ist, läuft die Uhrzeit weiter (gepuffert von der Batterie im Netzteil).

Uhrzeit-Speicherstellen (alle Angaben in BCD):

C 0980	1/100 Sekunden (\$00 bis \$99)
C 0981	Sekunden (\$00 bis \$59)
C 0982	Minuten (\$00 bis \$59)
C 0983	Stunden (\$00 bis \$23)
C 0984	Tag (\$01 bis \$31)
C 0985	Monat (\$01 bis \$12)
C 0986	Jahr (\$00 bis \$99)
C 0987	Wochentag (1 bis 7)

Die Steuerung der Echtzeituhr erfolgt über zwei Speicherstellen:

F D85	Uhr ein/aus (1 = ein)
F D86	Uhr stellen ein/aus (0 = stellen ein)

Stellen der Echtzeituhr (Uhr muß eingeschaltet sein, d.h. F D85 muß 1 sein):

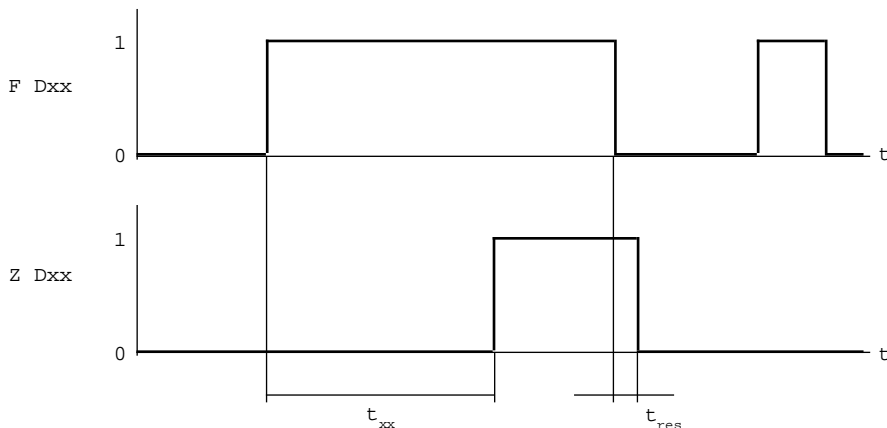
- Uhr stellen ein (F D86 löschen)
- Uhrzeit-Speicherstellen C 0980 bis C 0987 mit Uhrzeit/Datum laden
- F D86 wird beim nächsten Programmdurchlauf automatisch wieder gesetzt

## SOFTWAREZEITEN

Die MINICONTROL-Zentraleinheiten verfügen über 64 Softwarezeiten, die als Anzugsverzögerung arbeiten. Jede Softwarezeit besteht aus folgenden Adressen:

F Dxx	Freigabe (Starten) der Softwarezeit. Durch Beschreiben dieser Speicherstelle mit 1 wird die Softwarezeit xx (xx = 00 bis 63) gestartet. Diese Speicherstelle kann auch gelesen werden (z.B. um festzustellen, ob eine Softwarezeit gestartet ist, oder nicht).
Z Dxx	Ergebnis. Ist diese Speicherstelle 1, so ist die dazugehörige Softwarezeit abgelaufen. Diese Speicherstelle kann nur gelesen werden. Das Zurücksetzen erfolgt durch Löschen der Freigabe F Dxx.
Zxx    n"nn	Zeitdefinition. Mit der Anweisung Zxx wird die Dauer der Softwarezeit in Sekunden und 1/100 Sekunden festgelegt. Diese Anweisung muß immer durchlaufen werden, sie steht deshalb meist am Anfang des Anwenderprogrammes.

### Zeitlicher Ablauf:



Nach Start der Softwarezeit xx durch Beschreiben der Freigabeadresse F Dxx mit 1 und Ablauf der mit der Zeitdefinition Zxx eingestellten Zeit  $t_{xx}$  wird die Zeitadresse Z Dxx ebenfalls 1.

Nach dem Rücksetzen der Freigabeadresse F Dxx wird die Zeitadresse Z Dxx beim nächsten Durchlauf durch die Zeitdefinition Zxx zurückgesetzt. Die Rücksetzzeit  $t_{res}$  kann im ungünstigsten Fall einen Programmzyklus lang sein.

**Beispiel:** 5,5 Sekunden nach Betätigen eines Tasters (E 042) soll ein Motor (A 058) gestartet werden. Mit einem weiteren Taster (E 043) soll der Motor wieder gestoppt werden:

```

0000      Z10      5"50      Zeitdefinition
0001      LAD      N      E 042      Taster START
0002      PRS      M 100      Pos. Flanke von E 042
0003      EXO      M 100      Pos. Flanke von E 042
0004      RST      M 100      Pos. Flanke von E 042
0005      PRS      F D10      Start Motorverzögerung
0006      LAD      E 043      Taster STOP
0007      RST      F D10      Start Motorverzögerung
0008      LAD      Z D10      Motorverzögerung
0009      =      A 058      Motor
0010      END

```

Das selbe Programmbeispiel kann auch mit einem Kontaktplan gelöst werden:

```

0000      Z10      5"50      Zeitdefinition
0001      SPU      KOP1      Kontaktplan-Aufruf
0002      END

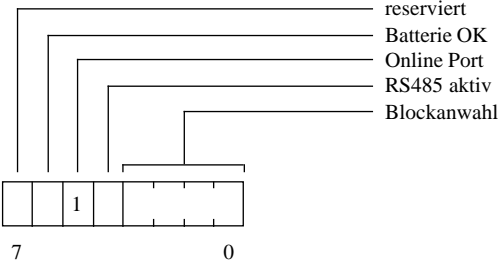
! E 042      M 100      F D10
00 --I I---+---I+I---+---+---+---+---+---+---+---(P)---
! M START      FLANKE      M VERZ.
!
! E 043      F D10
01 --I I---+---+---+---+---+---+---+---+---+---(R)---
! M STOP      M VERZ.
!
! Z D10      A 058
02 --I I---+---+---+---+---+---+---+---+---+---( )---
! MOT.EIN      MOTOR

```

Die Zeitdefinition Zxx muß bei jedem Programmdurchlauf genau ein mal durchlaufen werden. Wird sie nicht durchlaufen, so ist die Funktion der Softwarezeit nicht mehr gewährleistet, wird sie mehrmals je Programmzyklus durchlaufen, so ist die angegebene Zeit nicht korrekt.

Jede Softwarezeit belegt eine 8 Bit-Speicherstelle im Bereich von C 0800 bis C 0863, der als Vorteiler verwendet wird und eine weitere 8 Bit-Speicherstelle im Bereich von C 0900 bis C 0963 als Zähler. Die Zeitdefinition Zxx ist ein Softwareinterrupt, der ca. 0,5 ms dauert (bei Verwendung vieler Softwarezeiten Auswirkung auf die Programmzykluszeit beachten!).

## INPORT/OUTPORT ADRESSE \$3400

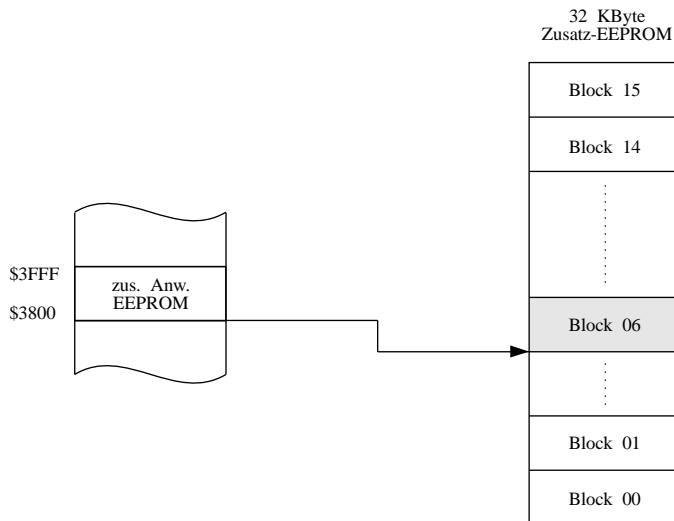
	
<b>reserviert</b>	Für spätere Anwendungen reserviert.
<b>Batterie OK</b>	<p>Dieses Bit dient zur Überwachung der Lithiumbatterie im Netzteil. Eine Auswertung kann auch über die Speicherstelle Z D99 erfolgen (siehe Abschnitt "BATTERIEKONTROLLE").</p> <p>0 ... Batterie leer (Spannung &lt; 2,60 V)  1 ... Batterie OK (Spannung &gt; 2,85 V)</p>
<b>Online Port</b>	Dieses Bit ist immer 1.
<b>RS485 aktiv</b>	<p>Dient zur Umschaltung zwischen TTY und RS485 Schnittstelle.</p> <p>0 ... TTY  1 ... RS485</p> <p>Nähere Beschreibung siehe Abschnitt "ANWENDER SCHNITTSTELLE".</p>
<b>Blockanwahl</b>	<p>Mit den Bits 0 - 3 wird der anzusprechende 2 KByte Speicherbereich des Zusatz-EEPROMs ausgewählt.</p> <p>0 ... Block 0  1 ... Block 1  :       :  :       :  F ... Block 15</p> <p>Nähere Beschreibung siehe Abschnitt "ZUSÄTZLICHES ANWENDER-EEPROM".</p>

## ZUSÄTZLICHES ANWENDER-EEPROM

Dieses EEPROM ist 32 KByte groß. Es ist in 16 Blöcke zu je 2 KByte unterteilt. Der gewünschte Block wird mit den ersten 4 Bits des Input/Output Bytes (Adresse \$3400) definiert.

**Beispiel:** Definition von Block 6 des Zusatz-EEPROMs.

LD	# \$3400	ERD mit \$3400 laden
DXR		Indexregister auf Adresse \$3400 setzen
LAD	I 000	Inhalt in ERA laden
UND	# %11110000	Bit 0 bis 3 löschen
OD	# 006	Block 6 definieren
=	I 000	ERA in Adresse \$3400 speichern



## DATENZUGRIFF AUF DAS ZUSÄTZLICHE ANWENDER-EEPROM

Der Anwender kann auf den selektierten Block des zusätzlichen Anwender-EEPROMs über die Adressen \$3800 bis \$3FFF zugreifen.



**DATEN LESEN**

**Zum Lesen von Daten aus einem Block wird das AWL-Makro DFEE verwendet:**

Übergabeparameter:	Quelle	...	Blocknummer in C 0881
			Offset zu Adresse \$3800 in C 0882&
	Ziel	...	Indexregister
	Datenlänge	...	in Ergebnisregister D
Rückgabeparameter:	Kein Fehler aufgetreten:		
	Carry = 0	...	Datentransfer OK
	Fehler aufgetreten:		
	Carry = 1	...	Die Summe von Offset und Datenlänge liegt außerhalb gültigem Bereich (> \$3FFF)
verwendete 8 Bit-Speicher:	C 0866&	...	Quelladresse
	C 0868&	...	Zieladresse
	C 0884&	...	aktuelle Datenlänge

```

DFEE  =D    C 0884 DATA 04          aktuelle Datenlänge
      =R    C 0868 & DEST 0
      LD    C 0882 DATA 02          Offset
      UND   # %00000111             begrenzen auf 2k Byte / Block
      +D    # $3800                  Startadresse EEPROM-Block
      =D    C 0866 & SOURCE 0
      +D    C 0884 DATA 04          aktuelle Datenlänge
      -D    # $3FFF
      J<=   DFEE0
*-----
      SEC                               Daten außerhalb gültigem Bereich
      RET
*-----
DFEE0 LD    # $3400                  Inport/Outport Adresse
      DXR
      LAD   C 0881 DATA 01
      UND   # %00001111             Blocknummer auf 0 ... F begrenzen
      =     C 0881 DATA 01
      LAD   I 000
      UND   # %11110000
      OD    C 0881 DATA 01
      =     I 000                    Block anwaehlen
*----
```

LD	C 0884 DATA 04	aktuelle Datenlänge
SRD		
JC0	DFE1	keine ungerade Datenlänge
=D	C 0884 DATA 04	aktuelle Datenlänge
LR	C 0866 & SOURCE 0	
LAD	I 000	erstes Byte lesen
IR		
=R	C 0866 & SOURCE	
LR	C 0868 & DEST 0	
=	I 000	erstes Byte speichern
IR		
=R	C 0868 & DEST 0	
LD	C 0884 DATA 04	aktuelle Datenlänge
*---		
DFE1	=D C 0884 DATA 04	aktuelle Datenlänge
SP0	DFE2	fertig
LR	C 0866 & SOURCE 0	
LD	I 000	Daten lesen
IR		
IR		
=R	C 0866 & SOURCE 0	
LR	C 0868 & DEST 0	
=D	I 000	Daten speichern
IR		
IR		
=R	C 0868 & DEST 0	
LD	C 0884 DATA 04	aktuelle Datenlänge
-D	# 00001	
SPI	DFE1	noch nicht alle Daten kopiert
*-----		
DFE2	CLC	Datentransfer OK
	RET	

**Beispiel:** Aus Block 4 werden ab der Adresse \$3A00 50 Bytes ausgelesen. Gespeichert werden die Daten ab C 2000.

LAD	# 004	Blocknummer 4
=	C 0881 DATA 00	
LD	# \$0200	Offset zu \$3800 (Quelladr. = \$3A00)
=D	C 0882 DATA 01	
LRK	C 2000 Zieladresse	
LD	# 00050	Datenlänge
SPU	DFEE	Daten aus Zusatz-EEPROM lesen
JC0	OK	
:		ERROR-Auswertung
:		

## DATEN SCHREIBEN

Beim Schreiben von Daten in ein EEPROM ist zu beachten, daß dies im Gegensatz zum Schreiben in einen 1 oder 8 Bit-Speicherbereich mit einer gewissen Verzögerung geschieht.

### **Zum Schreiben von Daten in einen Block wird das AWL-Makro DTEE verwendet:**

Das AWL-Makro DTEE eignet sich zum Programmieren von Parameterdaten, die sich während des Betriebes einer Anlage nicht ändern.

**ACHTUNG:** Wenn diese Daten auf das EEPROM geschrieben werden, wird das weitere Programm **nicht** bearbeitet!

Für das Beschreiben des EEPROMs während eines Programmdurchlaufes ist ein entsprechender Funktionsblock in Vorbereitung.

Übergabeparameter:	Quelle	...	Indexregister
	Ziel	...	Blocknummer in C 0881
			Offset zu Adresse \$3800 in C 0882&
	Datenlänge	...	in Ergebnisregister D
Rückgabeparameter:	Kein Fehler aufgetreten:		
	Carry = 0	...	Datentransfer OK
	C 0881	...	Blocknummer des nächsten freien Bytes
	C 0882&	...	Offset zu \$3800 des nächsten freien Bytes
	Fehler aufgetreten:		
	Carry = 1	...	Datentransfer fehlerhaft
	ERA	...	Fehlernummer:
			1 - Datenlänge größer als freier Speicher
			2 - EEPROM defekt
	C 0881	...	Blocknummer der defekten Speicherstelle
	C 0882&	...	Offset zu \$3800 der defekten Speicherstelle
verwendete 8 Bit-Speicher:	C 0880	...	Runtime-Zähler
	C 0884&	...	aktuelle Datenlänge
	C 0886&	...	Quell-Pointer

```

DTEE  =D   C 0884 DATA 04      aktuelle Datenlänge
      =R   C 0866 & SOURCE 0
      LD   C 0882 DATA 02      Offset
      UND  # %00000111         begrenzen auf 2k Byte / Block
      +D   # $3800              Startadresse EEPROM-Block
      =D   C 0882 DATA 02
      +D   C 0884 DATA 04
      -D   # $3FFF
      J<=  DTE5

*-----
      LAD  # 001                ERROR ... Datenlänge größer als
      SEC                                     freier Speicher
      RET

*-----
DTE5  LAD  C 0998 CYCLE TIME COUNTER
      =    C 0886 DATA 06
DTE3  LAD  C 0886 DATA 06
      =    C 0998 CYCLE TIME COUNTER

*
      LD   # $3400              Inport/Output Adresse
      DXR
      LAD  C 0881 DATA 01
      UND  # %00001111         Blocknummer auf 0 ... F begrenzen
      =    C 0881 DATA 01
      LAD  I 000
      UND  # %11110000
      OD   C 0881 DATA 01
      =    I 000                Block anwählen

*
      LR   C 0866 & SOURCE 0
      LAD  I 000                Daten lesen
      LR   C 0882 DATA 02
      =    I 000                akt. Kopierdaten abspeichern

*----
      ANS
      LD   # 01500
DTE0  -D   # 00001              Warteschleife
      SN0  DTE0
      AVS

*----
      CLR  C 0880 DATA 00      Runtimezaehler ruecksetzen
DTE1  LB   I 000                Daten von EEPROM mit aktuellen
      AVB                                     Kopierdaten vergleichen
      SP0  DTE2
      INC  C 0880 DATA 00      Runtimezaehler erhoehen
      LB   C 0880 DATA 00
      VB   # 200                mit Runtime MAX vergleichen
      SP<  DTE1

*-----
      LAD  # 002
      SEC                                     ERROR .... EEPROM defekt
      RET

*-----

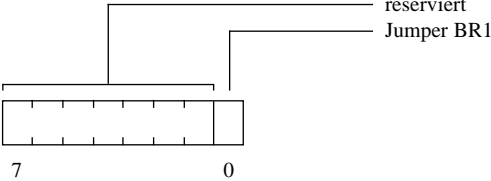
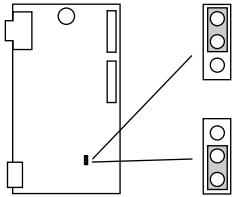
```

DTE2	LD	C 0882 DATA 02	MEM-Adresspointer erhoeuen
	+D	# 00001	
	=D	C 0882 DATA 02	
	LR	C 0866 & SOURCE 0	
	IR		
	=R	C 0866 & SOURCE 0	
	LD	C 0884 DATA 04	aktuelle Datenlänge
	-D	# 00001	alle Daten kopiert ?
	=D	C 0884 DATA 04	
	SP0	DTE4	
	SPI	DTE3	
DTE4	CLC		Datentransfer OK
	RET		

**Beispiel:** In Block 8 werden ab der Adresse \$3B00 40 Bytes geschrieben. Die zu schreibenden Daten sind ab der Speicherstelle C 2500 gespeichert.

LAD	# 008	Blocknummer 8
=	C 0881 DATA 00	
LD	# \$0300	Offset zu \$3800 (Zieladr. = \$3B00)
=D	C 0882 DATA 01	
LRK	C 2500 Quelladresse	
LD	# 00040	Datenlänge
SPU	DTEE	Daten in Zusatz-EEPROM schreiben
JC0	OK	
:		ERROR-Auswertung
:		

# INPORT ADRESSE \$3480

<div><p>reserviert</p><p>Jumper BR1</p><p>7 0</p></div>	
reserviert	Für spätere Anwendungen reserviert.
Jumper BR1	<p>In Steckplatz 0 kann ein Diagnosemodul betrieben werden (Anzeige von C 0000). Mit einem Schiebeschalter des Diagnosemodules wird der Eingang E 004 gesetzt. Während dem Einschalten wird dieser Eingang vom Betriebssystem überprüft.</p> <p>E 004 = 0 ... kein Diagnosemodul in Steckplatz 0 E 004 = 1 ... es befindet sich ein Diagnosemodul in Steckplatz 0 -&gt; Anzeige über die Ausgänge A 000 bis A 00F bedienen</p> <p>Durch neue Module (wie die Mischmodule MAEA und MAEB) kann es vorkommen, daß der Eingang E 004 während dem Einschalten gesetzt ist. Das Betriebssystem versucht nun über die Ausgänge A 000 bis A 00F die Anzeige des Diagnosemodules zu bedienen.</p> <p>Um zu vermeiden, daß die Ausgänge unbeabsichtigt gesetzt werden, befindet sich auf der Vorderseite der CPU-Platine ein Jumper:</p> <div><p>BR1 = 1 - Betrieb mit Diagnosemodul</p><p>BR1 = 0 - keine E 004 Abfrage (Auslieferungszustand)</p></div>

## RUNTIME-ÜBERWACHUNG

Mit der Runtime-Überwachung wird die maximal zulässige Programmzykluszeit von 100 ms überprüft. Ist ein Programmzyklus nach dieser Zeit noch nicht beendet, so wird das Anwenderprogramm gestoppt, und ein Software-Reset ausgelöst (alle Ausgänge werden zurückgesetzt). Ein Runtimefehler wird im Statustest des Programmiergerätes und durch Einschalten der Status-LED angezeigt.

## TIMERINTERRUPT-ROUTINEN

Unabhängig von der Länge des Anwenderprogrammes wird alle 10 ms ein Interrupt ausgelöst und die sogenannte Timerinterrupt-Routine ausgeführt. Diese Betriebssystemfunktion wird für Sicherheits- und Diagnosefunktionen sowie für die Generierung von Softwarezeiten, Uhrzeitfunktionen, Zeittakten und Zeitimpulsen verwendet.

Der Timerinterruptvektor (die Adresse der Timerinterrupt-Routine) steht in C 0972, 0973. Die Timerinterrupt-Zeit ist in C 0974, 0975 gespeichert (Einheit  $\mu\text{s}$ ). Timerinterrupt-Vektor und Timerinterrupt-Zeit dürfen vom Anwenderprogramm nicht geändert werden.

Zusätzlich zu den Betriebssystem-Funktionen kann der Anwender selbst einen oder zwei Programmteile zeitgesteuert ausführen lassen (User-Timerinterrupt-Routinen). Dazu werden die Timerinterrupt-Handler \$US1 und \$US2 verwendet. Die Parameter:

ERA	Gewünschtes Zeitintervall in ms
R	Anfangsadresse der User-Timerinterrupt-Routine

**Aufruf:**                      SPU    \$US1                      bzw.                      SPU    \$US2

Die User-Timerinterrupt-Routine wird mit RET abgeschlossen. Unabhängig vom gewählten Zeitintervall für die User-Timerinterrupt-Routine wird die Betriebssystem-Timerinterrupt-Routine alle 10 ms ausgeführt.

**ACHTUNG:**                      Timerinterrupt-Routinen werden nicht ausgeführt, wenn die SPS im HALT-Zustand ist.

Zu häufiges Aufrufen von langen Timerinterrupt-Routinen kann die Programmzykluszeit wesentlich verlängern und zu Systemstörungen führen. Die Summe der Ausführungszeiten beider Timerinterruptroutinen darf maximal 300  $\mu\text{s}$  betragen.

In Timerinterrupt-Routinen dürfen keine Betriebssystem-Mathematikroutinen verwendet werden.

Zum Ausschalten einer aktivierten User-Timerinterrupt-Routine wird ERA mit 0 geladen und der Interrupt-Handler (\$US1 oder \$US2) erneut aufgerufen.

**Beispiel:** Alle 3 ms soll der Zählerstand eines Abwärtszählers ausgelesen und mit 10000 verglichen werden. Bei Unterschreitung dieses Wertes soll ein Ausgang gesetzt werden. Der Timerinterrupt-Handler \$US1 wird nur ein mal in einer Initialisierungsroutine aufgerufen:

```

INIT  LAD    Z D64          First Scan
      SP0    INIR
      LAD    # 003          3 ms
      LRL    TEST          Adresse der Interrupt-Routine
      SPU    $US1
INIR  RET

TEST  SPU    READ          Zählerstand auslesen
      -D     # 10000        Vergleich mit 10000
      JC0    TESR
      SET    A 040          Zähler low !
TESR  RET

```

## FEHLERMELDUNGEN

Alle Zentraleinheiten sind mit umfangreichen Sicherheits- und Diagnosefunktionen ausgestattet (z.B. Programm-Checksumtest bei Power-on). Im Fehlerfall wird das Anwenderprogramm angehalten, die Status-LED eingeschaltet und ein Software-Reset ausgelöst, d.h. alle digitalen Ausgänge werden gelöscht, alle analogen Ausgänge werden auf 0 V bzw. 0 mA zurückgesetzt. Falls ein Programmiergerät angeschlossen ist, wird im Statustest eine Klartext-Fehlermeldung angezeigt (z.B. RUNTIME-FEHLER).

Die folgende Tabelle ist eine Übersicht über alle bei MINICONTROL Zentraleinheiten möglichen Fehlermeldungen:

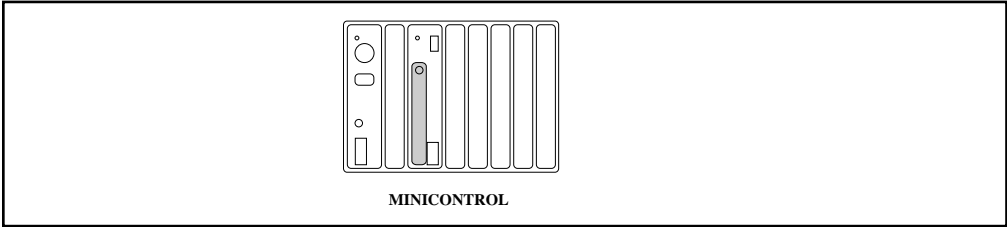


Bezeichnung	Beschreibung/Ursachen	Abhilfe
Übertragungsfehler bei Download	<p>Beim Übertragen eines Programmes vom Programmiergerät in die SPS (Download) tritt ein Fehler auf.</p> <p><b>Mögliche Ursachen:</b> Die Online-Verbindung zwischen PG und SPS wird durch starke, elektromagnetische Störungen beeinträchtigt</p>	<p>Programm erneut in die SPS übertragen. Im Wiederholungsfall wenn möglich Lichtleiteronlekabel (FOL) verwenden.</p>
Write Protect	<p>Dieser Fehler tritt nur im Zusammenhang mit EP05 EPROM-Modulen auf.</p> <p><b>Ursache:</b> Es wurde versucht, ein Programm mit RUN in ein EP05 EPROM-Modul in der Zentraleinheit zu übertragen</p>	<p>RAM-Programmspeichermodul verwenden.</p>
Checksum-Fehler nach RUN	<p>Ein mit RUN übertragenes Programm weist im RAM der SPS eine falsche Prüfsumme (Checksum) auf.</p> <p><b>Ursache:</b> Programmspeicher defekt.</p>	<p>Programm erneut übertragen, im Wiederholungsfall EE32 tauschen</p>
RAM zu klein	<p>Dieser Fehler tritt nur im Zusammenhang mit RA02 RAM-Modulen auf.</p> <p><b>Ursache:</b> Es wurde versucht, ein Programm, das auf 4k7 expandiert ist, in ein RA02-Modul zu übertragen.</p>	<p>Anderes Anwenderprogrammspeichermodul verwenden.</p>
Checksum-Fehler	<p>Die Prüfsumme (Checksum) des Anwenderprogrammes ist nach Reset oder Power-on falsch.</p> <p><b>Mögliche Ursachen:</b> Bei PROM-Programm PROM-Speicher defekt, bei RAM-Programm Batteriepufferung ausgefallen (leer oder defekt) oder Softwarefehler, der das Anwenderprogramm überschreibt.</p>	<p>Programm erneut übertragen. Im Wiederholungsfall Batteriepufferung überprüfen, Anwenderprogramm auf Softwarefehler untersuchen, Programmspeichermodul tauschen.</p>
Runtime-Fehler	<p>Die zulässige Programmzykluszeit von 100 ms wurde überschritten.</p> <p><b>Mögliche Ursachen:</b> Softwarefehler, zu viele Programmschleifen, Endlosschleife.</p>	<p>Programmfehler beheben.</p>

Bezeichnung	Beschreibung/Ursachen	Abhilfe
Pointer-Fehler	<p>Beim Checksumtest während Power-on wurde festgestellt, daß Betriebssystemvektoren nicht stimmen.</p> <p><b>Mögliche Ursachen:</b> siehe "Checksum-Fehler".</p>	Siehe "Checksum-Fehler".
Kommunikationsfehler	<p>Bei der Kommunikation zwischen dem Programmiergerät und der Zentraleinheit (RUN, Statustest) tritt ein Fehler auf.</p> <p><b>Mögliche Ursachen:</b> Die Onlineverbindung zwischen PG und SPS wird durch starke, elektromagnetische Störungen beeinträchtigt.</p>	Funktion wiederholen. Im Wiederholungsfall wenn möglich Lichtleiteronlinekabel (FOL) verwenden.
Store-Fehler	<p>Unzulässiger Schreibbefehl auf geschützte Speicherbereiche (ab \$C000).</p> <p><b>Mögliche Ursachen:</b> Fehler im Anwenderprogramm (Schreibbefehl mit indizierter Adressierung).</p>	Programmfehler beheben.
Stapelzeiger-Fehler	<p>Am Programm-Ende (END) steht der Stapelzeiger (Stackpointer) falsch.</p> <p><b>Mögliche Ursachen:</b> Fehler im Anwenderprogramm (Unterprogramm nicht mit RET abgeschlossen, Fehler bei Verwendung des System-Stacks zur Datenspeicherung).</p>	Programmfehler beheben.
Trap-Fehler	<p>Unbekannter Prozessorbefehl</p> <p><b>Mögliche Ursachen:</b> Fehler im Anwenderprogramm (z.B. Indizierter Sprung auf Datenbereich).</p>	Programmfehler beheben.
Interrupt-Fehler	<p>Durch unbefugten Zugriff auf Betriebssystem-Speicherbereiche (\$0000 bis \$0020) wurde ein nicht zulässiger Interrupt freigegeben und ausgelöst.</p> <p><b>Mögliche Ursachen:</b> Fehler im Anwenderprogramm (Schreibbefehl mit indizierter Adressierung).</p>	Programmfehler beheben.

# ANWENDERPROGRAMMSPEICHER

Der Anwenderprogrammspeicher wird zur Speicherung des Anwenderprogrammes benötigt. Er wird in den dafür vorgesehenen - grau markierten - Steckplatz der Zentraleinheit gesteckt und mit der Befestigungsschraube arretiert.



## BESTELLNUMMERN - BESTELLBEZEICHNUNGEN

<b>EE32 kombiniertes RAM/EEPROM-Anwenderprogrammspeichermodul</b> , 16 kByte RAM, 16 kByte EEPROM, für 4,7 k Anweisungen, WE/WP-Schalter (Schreibschutz für EEPROM), LED-Anzeige für Schreibzugriff	<b>EC EE32-0</b>
<b>EP05 EPROM-Anwenderprogrammspeichermodul</b> , 16 kByte EPROM für 4,7 k Anweisungen	<b>MC EP05-0</b>

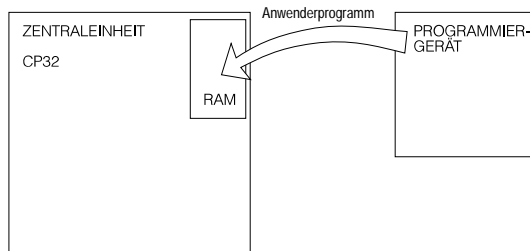
Beide MINICONTROL Anwenderprogrammspeichermodule können auch in den Zentraleinheiten CP40 (MULTICONTROL), CP41 (MIDICONTROL), NTCP3# (M264) sowie in den Peripherieprozessoren PP40 eingesetzt werden.

## INTERNES RAM

Das interne RAM ist **nur** zur Verwendung während der Inbetriebnahme oder zum Austesten von Programmen gedacht. Um das Programm nullspannungssicher zu speichern muß ein EE32 oder EP05 Speichermodul verwendet werden.

### Übertragen eines Anwenderprogrammes in die Zentraleinheit (RUN):

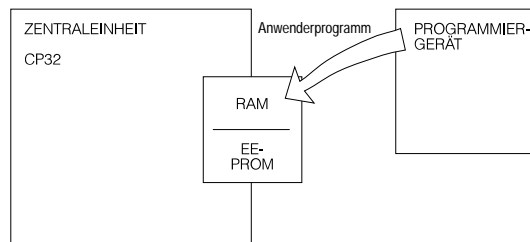
Wenn sich beim Übertragen eines Anwenderprogrammes vom Programmiergerät in die Zentraleinheit kein externer Anwenderprogrammspeicher (EE32 oder EP05) in der CPU befindet, wird dieses im internen RAM der CPU gespeichert und gestartet.



## EE32 - RAM/EEPROM ANWENDERPROGRAMM-SPEICHERMODUL

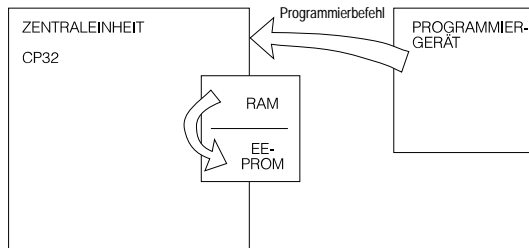
### Übertragen eines Anwenderprogrammes in die Zentraleinheit (RUN):

Beim Übertragen eines Anwenderprogrammes vom Programmiergerät in die Zentraleinheit wird dieses im RAM des EE32 gespeichert und gestartet, unabhängig davon, ob im EEPROM des EE32 ein anderes Programm gespeichert ist.



### Programmieren des EEPROM-Speichers:

Mit einem Befehl aus dem EEPROM-Menü des Programmiergerätes wird die Zentraleinheit veranlaßt, das Programm vom RAM ins EEPROM des EE32 zu programmieren. Das Programmieren des EEPROMs kann auch bei laufendem Anwenderprogramm erfolgen. Ein EEPROM-Programmspeicher muß nicht gelöscht werden, er wird einfach mit dem neuen Programm überschrieben. Während des Programmierens des EE32 darf die SPS nicht ausgeschaltet werden.



Der WE/WP-Schalter des EE32 muß während des Programmierens auf WE (Write Enable) stehen.

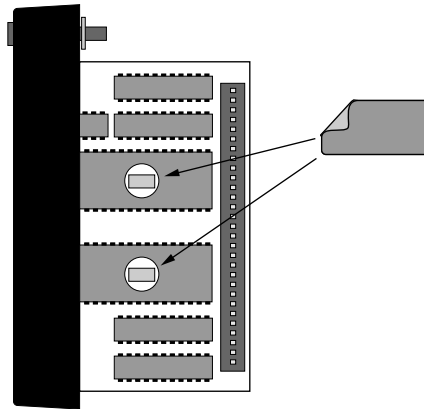
### Unterbrechungsfreies Übertragen eines Anwenderprogrammes in die Zentraleinheit (XFER):

Mit dem PG-Kommando XFER kann ein Anwenderprogramm in den RAM-Speicher des EE32 übertragen werden, ohne das im EEPROM-Speicher laufende Programm anzuhalten oder zu beeinflussen. Mit einem Befehl vom Programmiergerät kann zwischen den Programmen im RAM- und EEPROM-Speicher des EE32 umgeschaltet werden. Das Umschalten erfolgt synchron zum Programmzyklus, d.h. nach Absetzen des Umschaltbefehles wird der laufende Programmzyklus beendet und beim nächsten END auf den jeweils anderen Speicher umgeschaltet. Es erfolgt jedoch kein Reset, d.h. die Speicherstellen, die bei einem Software-Reset gelöscht werden (C 0000 bis C 0019), werden nicht verändert. Auch die First Scan-Speicherstelle C 0899 wird bei XFER und unterbrechungsfreiem Umschalten nicht gesetzt.

## EP05 - EPROM ANWENDERPROGRAMM-SPEICHERMODUL

Für die Programmierung des EP05 EPROM-Anwenderprogrammspeichers werden ein EPROM-Programmiergerät (Best.Nr. ECEP01-0) und ein EP05-Programmieradapter (Best.Nr. ECEPAD01-0) benötigt. Das Anwenderprogramm wird mit einem Befehl des B&R PROgrammierSYstemes als S-Record File abgespeichert und mit dem EPROM Programmer-Softwarepaket in den EPROM-Speicher programmiert. Das Softwarepaket ist im Lieferumfang des EPROM-Programmiergerätes enthalten.

EPROM-Speicher müssen vor dem Programmieren mit einer UV-Lampe gelöscht werden. Nach dem Programmieren sind die Löschenfenster lichtundurchlässig zu verkleben:



### Programm-Upload:

Anwenderprogramme können aus der MINICONTROL Zentraleinheit zurückgeladen werden, unabhängig davon, ob sie in einem EP05- oder EE32-Modul gespeichert sind. Das Zurückladen kann auch bei laufendem Anwenderprogramm erfolgen, in diesem Fall kann der Vorgang jedoch mehrere Minuten dauern.

Ein aus der Zentraleinheit zurückgeladenes Programm ist zwar lauffähig, im Programmiergerät stehen jedoch nicht mehr alle Informationen zur Verfügung. Es fehlen:

- Kontaktplanbilder
- Funktionsbausteinbilder
- Kommentare
- Klartextzuweisungen
- Datenformate in Tabellen

## EINSCHALTVERHALTEN (POWER-ON)

