POSITIONIER- UND BAHNSTEUERUNG MCU-3T / MCU-6 SOFTWARE-ÄNDERUNGEN UND ERWEITERUNGEN / SEW

Stand: 19.11.2002, Disk V2-52
Board-Revision: A

1 PCAP-Programmierung

1.1 BootFile, boot operating system file

TURBO PASCAL: function BootFile(var BootFileName:string; TpuBaseAddress: integer, softint:integer):integer;

C: int BootFile(char* BootFileName, int TpuBaseAddress);

BESCHREIBUNG: Diese Funktion dient zum Übertragen der Betriebssystemsoftware (rwtos.btl) auf die Steuerung. Das System wird zunächst zurückgesetzt. Anschließend wird die in *BootFileName* spezifizierte Datei (normalerweise rwtos.btl) auf die Steuerung geladen. In *TpuBaseAddress* wird die Basisadresse (Einstellung ab Werk: 300 hex) spezifiziert, unter welcher die Steuerung programmiert wird.

ANMERKUNG: Nach erfolgreichem Bootvorgang muß noch die Funktion InitMcuSystem(), InitMcuSystem2() oder InitMcuSystem3() aufgerufen werden, damit die Steuerung komplett initialisiert wird.

RÜCKGABEWERT: Die Funktion liefert folgende Rückgabewerte

Rückgabewert	Fehler-Beschreibung
0	kein Fehler, Bootvorgang erfolgreich abgeschlossen.
1	Die Schnittstelle konnte unter der angegeben Basisadresse (*TpuBaseAddress*) nicht eröffnet werden. Somit ist auch kein Zugriff auf die Steuerung möglich.
2	Der Rücksetzvorgang der Steuerung konnte nicht erfolgreich abgeschlossen werden.
3	Die in *BootFileName* spezifizierte Datei konnte nicht geöffnet werden.
4	Fehler beim Übertragen auf die Steuerung (falsche Daten).
5	Fehler beim Übertragen auf die Steuerung (Timeout).
6, 7	Timeout beim Warten auf Quittung der Steuerung.

1.2 InitMcuSystem2, initialise mcu system (2nd method)

TURBO PASCAL: function InitMcuSystem2(var tsrp:TSRP; TpuBaseAddress: integer, var SystemFileName: string; softint:integer):integer;

C: int InitMcuSystem2(struct TSRP *tsrp, int TpuBaseAddress, char *SystemFileName, int softint)

BESCHREIBUNG: Diese Funktion hat die gleiche Bedeutung wie *InitMcuSystem()*, mit der Ausnahme, daß die Parameter *SystemFileName* und *TpuBaseAddress* spezifiziert werden. Dabei enthält *SystemFileName* den Dateinamen der Systemdatei (normalerweise system.dat) incl. Pfad und *TpuBaseAddress* die Basisadresse der Steuerung (Einstellung ab Werk: 300 hex).

ANMERKUNG: siehe *InitMcuSystem()*

RÜCKGABEWERT: Die Funktion hat die gleichen Rückgabewerte wie die Funktion *InitMcuSystem()*.

1.3 InitMcuSystem3, initialise mcu system (3rd method)

TURBO PASCAL: function InitMcuSystem3(var tsrp:TSRP; var tosi:TOSI, TpuBaseAddress: integer, var SystemFileName: string; var BoardType: integer; softint:integer):integer;

C: int InitMcuSystem3(struct TSRP *tsrp, struct TOSI *tosi, int TpuBaseAddress, char *SystemFileName, int *BoardType, int softint)

BESCHREIBUNG: Diese Funktion hat die gleiche Bedeutung wie *InitMcuSystem()*, mit der Ausnahme, daß die Parameter *SystemFileName, tos, TpuBaseAddress* und *BoardType* spezifiziert werden. Dabei enthält *SystemFileName* den Dateinamen der Systemdatei (normalerweise system.dat) incl. Pfad und *TpuBaseAddress* die Basisadresse der Steuerung (Einstellung ab Werk: 300 hex).

ANMERKUNG: siehe *InitMcuSystem()*

RÜCKGABEWERT: Die Funktion hat die gleichen Rückgabewerte wie die Funktion *InitMcuSystem()*. Zusätzlich wird die Struktur *tosi* anhand der von der Steuerung zurückgelieferten Systeminformationen aktualisiert. Der Wert von *BoardType* gibt Aufschluß über den Steuerungstyp. Sofern *BoardType* den Wert 1 hat, handelt es sich um eine MCU-3T. Der Wert 2 hingegen spezifiziert eine MCU-6.

1.4 js, jog stop

TURBO PASCAL: procedure js(var as:AS; softint:integer);

C: void js(struct AS far *as, int softint);

BESCHREIBUNG: Die in AS gewählten Achskanäle werden mit der achsspezifischen Verzögerung *sdec* auf Geschwindigkeit 0 abgebremst und in Lageregelung gehalten. Bis zum Ende des Abbremsvorgangs ist das *pe*-Flag im *axst*-Register rückgesetzt. Die Verzögerung *sdec* kann mit Hilfe von Schreib- und Lese-Befehlen jederzeit gesetzt und abgefragt werden. Der Defaultwert wird im Hilfsprogramm *mcfg.exe* spezifiziert.

ANMERKUNG: Sofern dieser Befehl gleichzeitig für mehrere Achsen ausgeführt wird, können diese aufgrund der achsspezifischen Systemparameter zu unterschiedlichen Zeitpunkten die Zielpositionen erreichen [PHB / Kapitel 2.2.6.1].

1.5 lps, latch position synchronous

TURBO PASCAL: procedure lps(an:integer; mst:integer; softint:integer);

C: void lps(int an, int mst, int softint);

BESCHREIBUNG: Mit diesem Befehl kann ein Latch-Vorgang synchron zum Abtastzyklus des in *an* angewählten Achskanals ausgelöst werden. Nach dem Aufruf wird die Ist-Position *{rp}* nach jeweils *mst* Abtastintervallen zwischengespeichert. Sofern ein Latch-Vorgang stattgefunden hat, wird dies im *axst*-Register im Flag *lpsf* (Bit Nr. 16) angezeigt. Mit dem PCAP-Lesebefehl *rdlp()* oder dem SAP-Achsenqualifizierer *lp* kann die zwischengespeicherte Position ausgelesen werden. Das Auslesen löscht auch das *lpsf*-Flag im *axst*-Register.

ANMERKUNG: Der Befehl wird hauptsächlich beim Aufzeichnen von Konturen und Teach-In-Anwendungen verwendet, da er das Aufzeichnen von Positionsdaten in Echtzeit von einer oder mehreren Achsen ermöglicht. Typische Werte für *mst* sind 10..100 Abtastintervalle (-> 12.8ms..128.0ms). Der genaue Wert hängt jedoch von der Verarbeitungsgeschwindigkeit der jeweiligen Applikation ab.

1.6 ms, motion stop

TURBO PASCAL: procedure ms(var as:AS; softint:integer);

C: void ms(struct AS far *as, int softint);

BESCHREIBUNG: Die in *AS* gewählten Achskanäle werden mit der zur Zeit gültigen Bahnbeschleunigung bzw. Achsenverzögerung auf Geschwindigkeit 0 abgebremst und in Lageregelung gehalten. Bis zum Ende des Abbremsvorgangs ist das *pe*-Flag im *axst*-Register rückgesetzt. Der Richtungsvektor einer evtl. gerade ablaufenden Interpolation wird durch diesen Befehl nicht verändert. Falls die angewählten Achsen gerade einen Kreis abfahren, erfolgt die Abbremsung auf der Kreisbahn mit der angegebenen Bahnbeschleunigung.
Achsen, die mit einer Endgeschwindigkeit verfahren, werden mit der achsspezifischen Verzögerung *sdec* auf Geschwindigkeit 0 abgebremst

ANMERKUNG: Nicht gemeinsam interpolierende Achsen können den Zielpunkt zu unterschiedlichen Zeitpunkten erreichen.

1.7 rasms, reset ASM-2003 status register (-> PHB / 4.4.29)

Dieser Abschnitt gilt nur im Zusammenhang mit der Positioniersteuerung MCU-6.

TURBO PASCAL: procedure rasms(var as:AS; softint:integer);

C: void rasms(struct AS far *as, int softint);

BESCHREIBUNG: Mit diesem Befehl können verschiedene Fehlerflags im ASM-2003-Status-Register *asms* rückgesetzt werden. Im Detail sind dies die Fehlerbits 0, 1, 2, 3, 7 - *lerrtotx*, *lerrtotx*, *lerrd, reserr* und *dc*. Das Rücksetzen sollte nur in Ausnahmesituationen, z.B. in einer Fehlerüberwachungsroutine, ausgeführt werden.

ANMERKUNG: Bei verschiedenen ASM-2003-Firmware-Revisionen wird auch die rote LED [IHB, Kapitel 4.2] zurückgesetzt.

1.8 rdasms, read ASM-2003 status (-> PHB / 4.4.29)

Dieser Abschnitt gilt nur im Zusammenhang mit der Positioniersteuerung MCU-6.
Das *asms*-Register wurde mit folgenden Flags ergänzt.

Tabelle 14 [Ergänzung PHB, 4.4.29]: Bitkodierter Aufbau des asms-Wortes

Bit-Nr.	Funktion
5	*ope*: Das *ope*-Flag zeigt, daß ein System-Fehler auf dem Options-Print der Anschaltbaugruppe erkannt wurde. Derzeit wird der Fehler nur von der SSI- und OPMF-2001-Optionskarte erzeugt.
6	*options*: Sofern die Firmware der ASM-2003 Baugruppe Optionen unterstützt, ist das Flag *options* permanent auf 1 gesetzt. Folgende Optionen werden unterstützt: Echtzeitlatchen, SSI-Drehgeberverarbeitung und A/D-Umsetzung.
7	*dc*: Das *dc*-Flag (disconnect), zeigt an, daß die Kommunikation zwischen TPU-6002 und ASM-2003-Anschaltbaugruppe für mehr als 50 Abtastintervalle (i. A. 64ms) ausgefallen war. In diesem Fall durchläuft die rw_TOS-Firmware eine interne Reset-Prozedur. Hierbei werden u.a. die anstehenden Bewegungskommandos der betroffenen Achskanäle verworfen und deren Lagelregelkreise geöffnet.

1.9 rdaxst, read axis status (-> PHB / 4.4.31)

Neu im *axst*-Register ist das Flag *lpsf* Bit Nr. 16. Dieses zeigt an, daß ein Latch-Vorgang synchron zum Abtastzyklus stattgefunden hat [Kapitel 1.1], oder ein mit LP-Funktion projektierter Digital-Eingang aktiviert wurde [Kapitel 4.2].

1.10 rddv, read desired velocity

TURBO PASCAL: procedure rddv(var tsrp:TSRP; softint:integer);

C: void rddv(struct TSRP far *tsrp, int softint);

TSRP-KOMPONENTEN: TSRP[n].dv

BESCHREIBUNG: Diese Funktion liefert die achsspezifische Soll-Geschwindigkeit des MCU-6-Profilgenerators zurück. Im Idealfall entspricht der eingelesene Wert der tatsächlichen Achsgeschwindigkeit (Ist-Geschwindigkeit).

RÜCKGABEWERT: Nach Ausführen der Funktion, steht die Sollgeschwindigkeit im Register *dv* in der achsspezifischen Geschwindigkeitseinheit zur Verfügung.

ANMERKUNG: Die Sollgeschwindigkeit kann nur durch entsprechende Verfahrbefehle beeinflußt werden.

1.11 rdlp, read latched position

TURBO PASCAL: procedure rdlp(var tsrp:TSRP; softint:integer);

C: void rdlp(struct TSRP far *tsrp, int softint);

TSRP-KOMPONENTEN: TSRP[n].lp

BESCHREIBUNG: Diese Funktion liefert die achsspezifische Latch-Position zurück. Der Latch-Vorgang kann durch verschiedene Mechanismen ausgelöst werden:
1. Beim Aktivieren eines mit LP-Funktion projektierten Eingangs. Hierbei beträgt die maximale Verzögerungszeit zwei Abtastintervalle (2.56ms). Ein neuer Latch-Vorgang wird erst nach disaktivieren des Latcheingangs ermöglicht.
2. Sofern zuvor ein *lps()*-PCAP-Kommando [Kapitel 1.1] ausgeführt wurde und die dort im Parameter *mst* spezifizierte Verzögerung abgelaufen ist.

3. In Echtzeit (max. 20µs Verzögerung) mit einer Spezial-ASM-2003-Firmware und Beschaltung des NMI-Digital-Eingangs. Ein neuer Latch-Vorgang wird erst nach disaktivieren des Latcheingangs ermöglicht.
Bei allen Methoden wird die Ist-Position {*rp}* der Motorachse zwischengespeichert.

RÜCKGABEWERT: Nach Ausführen der Funktion, steht die Latchposition im Register *lp* in der achsspezifischen Positionseinheit zur Verfügung.
Die Priorität der drei Methoden ist gleichbedeutend mit der Reihenfolge der Auflistung, d.h. Echtzeit-Latchen hat die höchste Priorität.

ANMERKUNG: PCAP-Befehl *wrlp()*

1.12 rdsdec, read stop deceleration

TURBO PASCAL: procedure rdsdec(var tsrp:TSRP; softint:integer);

C: void rdsdec(struct TSRP far *tsrp, int softint);

TSRP-KOMPONENTEN: TSRP[n].sdec

BESCHREIBUNG: Mit diesem Befehl kann die achsspezifische Stopverzögerung *sdec* eingelesen werden. Der Defaultwert wird mit Hilfe des TOOLSET Programms *mcfg.exe* festgelegt.

RÜCKGABEWERT: Nach Ausführung des Befehls steht die Stopverzögerung im Feld *sdec* zur Verfügung. Der Wert wird in der achsspezifischen Beschleunigungseinheit zurückgeliefert.

ANMERKUNG: Die Stopverzögerung kann unter anderem mit dem PCAP-Befehl *wrsdec()* jederzeit neu gesetzt werden.

1.13 ssf, Spool-Special-Function

C: void ssf(int axis, int command, int value);

BESCHREIBUNG: Dieses Kommando ermöglicht dem Anwender auch andere Kommandos als Verfahrbefehle im Spooler einzutragen. Mit dem Parameter *command* wird das auszuführende Kommando eingetragen. Der Wert *value* wird als Parameter bei der in axis angegebenen Achse eingetragen.
Folgende Kommandos sind verfügbar:

0 .. 99	CI-Variable mit *Value* beschreiben
1000	Spoolerabarbeitung anhalten, diese Anweisung wird nur dann ausgeführt, wenn die Zielgeschwindigkeit des letzten eingetragenen Profils gleich 0 ist.
1001	Digitale Ausgänge setzen, die zu setzenden Ausgäng werden in *Value* bitweise angegeben.
1002	Digitale Ausgänge rücksetzen, die rückzusetzenden Ausgänge werden in *Value* bitweise angegeben.
1003	Spoolerabarbeitung anhalten für die in *Value* angegebene Zeit. Zeiteinheit ist 64 µs. Die tatsächliche Wartezeit wird auf Vielfache der Abtastzeit abgerundet. Diese Anweisung wird nur dann ausgeführt, wenn die Zielgeschwindigkeit des letzten eingetragenen Profils gleich 0 ist. Der Wartevorgang kann z.B.mit der Anweisung SSMS vorzeitig beendet werden.
1004	Spoolerabarbeitung anhalten bis die Eingänge aktiv sind, die in *Value* angegeben sind. Diese Anweisung wird nur dann ausgeführt, wenn die Zielgeschwindigkeit des letzten eingetragenen Profils gleich 0 ist. Der Wartevorgang kann z.B. mit der Anweisung SSMS vorzeitig beendet werden.

BEISPIELE:

```
ssf(A1, 1, 999);          // CI1 mit 999 beschreiben
ssf(A1, 1001, 1);         // Ausgang O1 bei Achse 1 setzen

ssf(A1, 1002, 1);         // Ausgang O1 bei Achse 1 rücksetzen
```

1.14 wrcd, write common double (Änderung)

Der Wertebereich von *ndx* [PHB / Kapitel 4.4.72] ist jetzt 0 bis 99.

1.15 wrci, write common integer (Änderung)

Der Wertebereich von *ndx* [PHB / Kapitel 4.4.72] ist jetzt 0 bis 99.

1.16 wrlp, write latched position

TURBO PASCAL:	procedure wrlp(var tsrp:TSRP; softint:integer);
C:	void wrlp(struct TSRP far *tsrp, int softint);
TSRP-KOMPONENTEN:	TSRP[n].lp
BESCHREIBUNG:	Dieser Befehl setzt die achsspezifische Latchposition auf den in *lp* gesetzten Wert. Die Wertangabe erfolgt in der achsspezifischen Positionseinheit.
ANMERKUNG:	PCAP-Befehl *rdlp()*

1.17 wrsdec, write stop deceleration

TURBO PASCAL:	procedure wrsdec(var tsrp:TSRP; softint:integer);
C:	void wrsdec(struct TSRP far *tsrp, int softint);
TSRP-KOMPONENTEN:	TSRP[n].sdec
BESCHREIBUNG:	Mit diesem Befehl wird die achsspezifische Stopverzögerung *sdec* für den PCAP-Befehl *js()* [Kapitel 1.2], SAP-Befehl *JS()* [Kapitel 2.3], die mit SMD-projektierten Software-Endlagen [Kapitel 4.1] und die mit LSL_SMD bzw. LSR_SMD projektierten Digital-Eingänge [Kapitel 4.2] gesetzt. Sofern *wrsdec()* nicht zur Ausführung kommt, wird mit dem im TOOLSET-Programm *mcfg.exe* festgelegten Systemparameter gearbeitet. Der Systemparameter kann zu jedem beliebigen Zeitpunkt überschrieben werden.
ANMERKUNG:	Der aktuell gesetzte Wert von *sdec* kann mit dem PCAP-Befehl *rdsdec()* gelesen werden. Kapitel 4.1 und 4.2

2 Standalone-Applikations-Programmierung

2.1 System-Parameter

2.1.1 Common Integer / Double

Es sind nun 100 Common Integer Variable (CI0..CI99) und 100 Common Double Variable (CD0..CD99) definiert.

Ergänzung zu Tabelle 29 (PHB): *rw_SymPas* vordefinierte System-Parameter

Name	Typ	Kurzwortbedeutung	Funktion
SSFP	Integer	Spool-Special-Function-Parameter	Funktionsparameter für Spezialfunktionen in der Spooler-Betriebsart. Weitere Informationen beim SAP-Kommando *SSF*.
PN1 PN2 PN3	double	Plane-Normal	Flächen-Normale für MCA3D Kommando. Weitere Informationen beim Kommando MCA3D.

2.1.2 *OSVERSION* (Betriebssystem–Versionsinformation)

Mit dem vordefinierten Systemparameter *OSVERSION* (Typ integer) kann die aktuelle Betriebssystemversionsnummer des *rwtos.btl*-Files zur Laufzeit in einem SAP-Programm abgefragt werden. Die Versionsnummer ist in eine Haupt und eine Nebennummer gegliedert, wobei die Hauptnummer in 100er Schritten und die Nebennummer in 10er Schritten hoch gezählt wird. Die Versionsnummer 101 z.B. bedeutet dann Hauptnummer 1 und Nebennummer 01.

Beispiel:

```
CI1 := OSVERSION;        // aktuelle Versionsnummer der Variablen
                         // CI1zuweisen
```

2.2 Semantische Grammatik

2.2.1 Typen-Deklaration

2.2.1.1 Axis-Typ

Als neuer Variablentyp wurde der Typ AXIS eingeführt. Mit Hilfe dieser Deklaration können ab sofort auch variable Achsen definiert werden. Dies ist insbesondere für die Gestaltung von mehrfach verwendeten Unterprogrammen (Prozeduren) hilfreich in denen immer wiederkehrende Aktionen für verschiedene Achsen ausgeführt werden sollen. [Kapitel 2.3]

Beispiel:

```
var
    VA : AXIS;                      // variable Achse mit Namen VA
    VA.an := 0;                     // Achsennummer 0 an VA zuweisen (wichtig!)
    ol(VA);                         // open loop von Achse 0
    for VA.an := 0 to 5 do  cl(VA); // close loop Achse 0 .. 5
```

Anmerkung: Auch die Achsnummer der vordefinierten Achsenspezifizierer (A1 .. An), welche bei den Systemparametern definiert sind, können auf diese Weise neu vergeben werden. Dies kann jedoch schnell zu einer unübersichtlichen und fehlerhaften Programmierung führen. Sobald mit variablen Achsen gearbeitet werden soll, empfiehlt es sich entsprechende Variablen mit entsprechendem Symbolcharakter zu verwenden.

2.3 Achsen-Qualifizierer an

Der Achsenqualifizierer *an* (Typ integer) ist neu und beinhaltet die Achsnummer des entsprechenden Achsenbezeichners. Der Qualifizierer wurde im Zusammenhang der neu verfügbaren „variablen“ Achsnamen eingeführt. [Kapitel 2.2.1.1]

2.4 Achsen-Qualifizierer asms (->PHB / 6.3.3)

Der Achsenqualifizierer *asms* wurde mit den Flags *ope* , *options* und *dc* erweitert. Informationen hierzu sind im Kapitel 1.5 enthalten.

2.5 Achsen-Qualifizierer axst (->PHB / 6.3.3)

Der Achsenqualifizierer *axst* wurde mit dem *lpsf*-Flag erweitert. Informationen hierzu sind in den Kapiteln 1.6 und 4.3 enthalten.

Neu hinzugekommen ist der Achsenqualifizierer *dv, lp* und *sdec*. Informationen hierzu sind in den Kapiteln 1.7, 1.8, 1.9 und 1.12 enthalten.

2.6 SAP-Befehls-Referenzliste *rw_SymPas*

2.6.1 JS, jog stop

FUNKTIONSPARAMETER: *Spec*

SYSTEMPARAMETER: Qualifizierer: *sdec*

BESCHREIBUNG: Die Beschreibung erfolgt beim PCAP-Befehl *js()*.

Beispiel: *JS(A1);*

2.6.2 MCA3D, move circular absolute three-dimendional

FUNKTIONSPARAMETER: *Spec*, *Po*s

SYSTEMPARAMETER: *TRAC, TRVL, TRTVL, PHI, PN1, PN2, PN3*

BESCHREIBUNG: Dieser Befehl bewirkt die zirkulare Interpolation der drei spezifizierten Achskanäle. Bezüglich der Achsenauswahl gibt es keine Einschränkungen. Die Kreisinterpolation wird auf Basis eines Trapez-Drehzahl-Profiles, d.h. unter Berücksichtigung von Maximalbeschleunigung und Maximalgeschwindigkeit, durchgeführt. Als Interpolationsparameter werden die Bahnbeschleunigung tr*ac*, die Bahngeschwindigkeit tr*vl* und die Bahnzielgeschwindigkeit tr*tvl*. Die angegebenen Koordinaten spezifizieren den Kreismittelpunkt im Absolutmaßsystem. Dabei ist die Reihenfolge der Parameter massgebend.
Der Kreis kann in einer beliebigen Ebene abgefahren werden, welche durch die Flächen-Normale in PN1, PN2 und PN3 spezifiziert wird. Die aktuellen Startkoordinaten liegen immer in der angegebenen Ebene.
Der Winkel *phi* spezifiziert den abzufahrenden Verfahrwinkel mit der Einheit Grad. Die Drehrichtung wird durch das Vorzeichen der Winkelgröße festgelegt. Positive Werte bedeuten, Drehrichtung im Uhrzeigersinn und negative Werte Drehrichtung im Gegenuhrzeigersinn. Der Verfahrwinkelbereich ist nicht auf bestimmte Grenzen fixiert, d.h. es können auch Teil- oder Vielfach-Kreise abgefahren werden.

BEISPIEL:
```
MCA3D(A1 := 50.0, A2 := 0.0, A3 := 0.0, PN1 = 1.0, PN2 =0. 0, PN3 = 1.0,
PHI := 720.0);          // Kreis 45 Grad um A2 gedreht
```

2.6.3 MCA3DW, move circular absolute three-dimendional waiting

FUNKTIONSPARAMETER: *Spec*, *Po*s

SYSTEMPARAMETER: *TRAC, TRVL, TRTVL, PHI, PN1, PN2, PN3*

REFERENZEN: SAP-Befehl *mca3d()*

BESCHREIBUNG: Dieser Befehl ist identisch mit dem SAP-Befehl *MCA3D()*, jedoch wird hier zusätzlich das Profilende der beiden beteiligten Achsen abgewartet.

2.6.4 MS, motion stop

FUNKTIONSPARAMETER: *Spec*

SYSTEMPARAMETER: keine

BESCHREIBUNG: Die Beschreibung erfolgt beim PCAP-Befehl *ms()*.

BEISPIEL:
```
MS(A1, A2);
```

2.6.5 RAMS, reset ASM-2003 status register

Dieser Abschnitt gilt nur im Zusammenhang mit der Positioniersteuerung MCU-6.

FUNKTIONSPARAMETER: *Spec*

SYSTEMPARAMETER: keine

BESCHREIBUNG: Die Beschreibung erfolgt beim PCAP-Befehl *rams()* [Kapitel 1.4].

BEISPIEL: *RASM(A1, A2);*

2.6.6 SSF, Spool-Special-Function

FUNKTIONSPARAMETER: *Spec*, *Value*

SYSTEMPARAMETER: *SSFP*

SIMULTANFUNKTION: ja

BESCHREIBUNG: Dieses Kommando ermöglicht dem Anwender auch andere Kommandos als Verfahrbefehle im Spooler einzutragen. In der Systemvariable *SSFP* wird das auszuführende Kommando eingetragen. Der Wert *Value* wird als Parameter bei der jeweiligen Achse eingetragen.
Folgende Kommandos sind verfügbar:

0 .. 99	CI-Variable mit *Value* beschreiben
1000	Spoolerabarbeitung anhalten, diese Anweisung wird nur dann ausgeführt, wenn die Zielgeschwindigkeit des letzten eingetragenen Profils gleich 0 ist.
1001	Digitale Ausgänge setzen, die zu setzenden Ausgäng werden in *Value* bitweise angegeben.
1002	Digitale Ausgänge rücksetzen, die rückzusetzenden Ausgänge werden in *Value* bitweise angegeben.
1003	Spoolerabarbeitung anhalten für die in *Value* angegebene Zeit. Zeiteinheit ist 64 µs. Die tatsächliche Wartezeit wird auf Vielfache der Abtastzeit abgerundet. Diese Anweisung wird nur dann ausgeführt, wenn die Zielgeschwindigkeit des letzten eingetragenen Profils gleich 0 ist. Der Wartevorgang kann z.B.mit der Anweisung SSMS vorzeitig beendet werden.
1004	Spoolerabarbeitung anhalten bis die Eingänge aktiv sind, die in *Value* angegeben sind. Diese Anweisung wird nur dann ausgeführt, wenn die Zielgeschwindigkeit des letzten eingetragenen Profils gleich 0 ist. Der Wartevorgang kann z.B. mit der Anweisung SSMS vorzeitig beendet werden.

BEISPIEL:

```
SSF(A1:=999, SSFP = 1);          // CI1 mit 999 beschreiben
SSF(A1:=1, A4:=2, SSFP := 1001); // O1 bei Achse 1 und O2 bei Achse 4
                                 // setzen

SSF(A1:=0, A2:=0, A3:=0, SSFP:=1000); // Spoolerhalt bei A1, A2 und A3
```

3 Hilfsprogramm mcbt.exe

3.1 Parameter für das Hilfsprogramm mcbt.exe

Das Hilfsprogramm *mcbt.exe* [BHB/4.1.5] dient zum Laden der Betriebssystem-Software *rwtos.btl* auf die TPU-6002. Weiterhin ist es möglich einen Configurations-Check bzw. das Neu-Speichern von Systemdaten [BHB / 4.3.10] innerhalb dieses Programms auszuführen.
Dies kann typischerweise in der AUTOEXEC.BAT durchgeführt werden.

Der Aufruf ist wie folgt vorzunehmen:

```
mcbt {/option /option ...}
```

```
option = BB, BC, BD, BH, BR n und BS
```

Nachfolgend werden die einzelnen Optionen in alphabetischer Reihenfolge beschrieben:

/BB *filename*	Das in *filename* spezifizierte File wird an die TPU-6002 übertragen. Der Suchvorgang für *filename* ist dabei wie folgt: zuerst wird im aktuellen Arbeitsverzeichnis gesucht und anschließend in den in PATH spezifizierten Directorys. Sofern diese Option nicht benutzt wird, wird das File *rwtos.btl* übertragen.
/BC	Nach dem Übertragen der Betriebssystem-Software *rwtos.btl* wird die Systemdatei *system.dat* auf die TPU-6002 übertragen. Im Anschluß erfolgt ein Configurationscheck auf alle im Antriebssystem vorhanden Achsen.
/BD	Bei auftretenden Fehlern während dem Lade-Vorgang werden ausführliche Fehlertexte am Bildschirm ausgegeben.
/BH	Mit dieser Option werden alle verfügbaren Kommando-Optionen am Bildschirm angezeigt.
/BR *n*	Mit dieser Option kann mit der Variablen *n* spezifiziert werden, wie oft die Optionen BC oder BS im Fehlerfall wiederholt werden sollen.
/BS	BS speichert die in der Systemdatei *system.dat* enthaltenen Daten. Sofern diese Option zusammen mit der Option BC aufgerufen wird, wird der Speichervorgang nur im Fehlerfall von BC ausgeführt.

```
Beispiel:        mcbt /BC /BS /BR 5
```

Das oben gezeigte Beispiel sucht zunächst die Betriebssystem-Software *rwtos.btl* im aktuellen Arbeitsverzeichnis bzw. in der Pfad-Umgebung. Diese Datei wird dann auf die TPU-6002 übertragen. Im Anschluß wird die Systemdatei *system.dat* auf die TPU-6002 übertragen und ein Configurationscheck ausgeführt. Sollte ein Konfigurationsfehler festgestellt werden, so wird der Check im Fehlerfall bis zu 5 mal wiederholt. Ist dann immer noch ein Fehler vorhanden, werden die neuen Daten auf das System übertragen und gespeichert. Auch dieser Vorgang wird im Fehlerfall bis zu 5mal wiederholt.

3.2 Rückgabewert des Hilfsprogramms mcbt.exe

Das Hilfsprogramm mcbt.exe kann folgende Fehlernummern zurückliefern:

Fehlernummer	Beschreibung
0	kein Fehler aufgetreten
1	Ungültiger Funktions-Code beim Aufruf von txbf()
2	Datei nicht gefunden beim Aufruf von txbf()
3	Pfad nicht gefunden beim Aufruf von txbf()
4	Zu viele Dateien offen beim Aufruf von txbf();
5	Zugriff verweigert beim Aufruf von txbf()
6	Ungültiges File-Handle beim Aufruf von txbf()
12	Ungültiger Zugriff beim Aufruf von txbf()
20	CNC-File zu groß beim Aufruf von txbf()
21	Ungültiger File-Typ beim Aufruf von txbf()
25	TSR-Treiber
26	Falscher TSR-Treiber
27	Falsche rw_TOS-Software
30	Max. erlaubte Länge des Boot-File-Namen überschritten
31	Boot-Filename erwartet
32	Ungültige Anzahl von Wiederholungen
40	Kommando-Zeilen-Länge überschritten
41	Systemdatei *system.dat* nicht gefunden
42	Interner spwanvp()-Fehler
43	Boot-Fehler im Programm *iserver.exe*
50	ASM-2003 Timeout-Fehler
100	Configurations-Fehler
alle anderen	ungültig

Diese Fehlernummern sind in der Environment-Variablen ERRORLEVEL gespeichert und können beispielsweise in der AUTOEXEC.BAT ausgewertet werden.
Eine Fehlerbeschreibung kann auch mit der Option /BD am Bildschirm angezeigt werden.

4 Hilfsprogramm mcfg.exe

4.1 Software-Endlagen (-> BHB / 4.3.2.11)

Zu den bereits im BHB/Kapitel 4.3.2.11 beschriebenen Spezialfunktionen ist der Funktionstyp *SMD* (Stop Motor Deceleration) hinzugekommen. Beim Ansprechen dieser Software-Endlage wird die Achse automatisch mit der achsspezifischen Verzögerung {*sdec}* auf Geschwindigkeit 0 abgebremst und dann in Lageregelung gehalten. Ein weiteres Verfahren über die Endlage hinaus wird verhindert. Wenn der Positionssollwert die Endlagenposition unterschreitet, wird der Endschalterzustand wieder aufgehoben. Dieser Funktionstyp wird insbesondere bei unterlagerten Drehzahlregelkreisen und Schrittmotorantrieben bevorzugt.

4.2 ASM-2003-Digitaleingänge (-> BHB / 4.3.3.1)

Zu den bereits im BHB/Kapitel 4.3.3.1 beschriebenen Spezialfunktionen sind drei weitere Funktionstypen hinzugekommen.

LSL_SMD (Limit Switch Left Stop Motor (with) Deceleration) Die mit dieser Funktion projektierten Eingänge funktionieren als linke Hardware-Endschalter. Beim Ansprechen dieses Eingangs wird die Achse automatisch mit der achsspezifischen Verzögerung {*sdec}* auf Geschwindigkeit 0 abgebremst und dann in Lageregelung gehalten. Ein weiteres Verfahren über die Endlage hinaus wird verhindert. Das Ansprechen des Endschalters erfolgt üblicherweise beim Verfahren in negative Richtung und Überschreiten der entsprechenden Endlage. Wenn der Positionssollwert die Position unterschreitet, wird der Endschalterzustand wieder aufgehoben. Dieser Funktionstyp wird insbesondere bei unterlagerten Drehzahlregelkreisen und Schrittmotorantrieben bevorzugt.
Anmerkung: Die Verzögerung *sdec* muß so bestimmt werden, daß ein sicheres Anhalten des Antriebes gewährleistet ist, ohne daß die Motorachse in eine mechanische Begrenzung läuft und dadurch evt. Schäden verursacht. Zur Absicherung des Antriebes sollten zusätzlich Hardware-Begrenzungsschalter eingesetzt werden, welche die Leistungsverstärker nur in die erlaubte Verfahrrichtung freischalten.

LSR_SMD (Limit Switch Right Stop Motor (with) Deceleration) Die Funktionsweise ist mit LSL_SMD identisch, bis auf den Unterschied, daß dieser Endschalter für Endlage rechts projektiert wird.

LP (Latch Position) Beim Aktivieren dieses Eingangs wird die Ist-Position *{rp}* der entsprechenden Motor-Achse zwischengespeichert. Sofern ein Latch-Vorgang ausgelöst wurde, ist das *lpsf*-Flag des *axst*-Registers gesetzt. Dann kann mit dem PCAP-Befehl *rdlp()* [Kapitel 1.8] oder über den SAP-Achsenqualifizierer *lp* die zwischengespeicherte Position eingelesen werden. Das *lpsf*-Flag wird durch den Einlesevorgang automatisch gelöscht. Die maximale Verzögerung des Latch-Vorganges beträgt 2 Abtastintervalle (2.56ms).

4.3 Bitinformationen des Achsenstatus-Register (-> BHB / 4.3.8.4)

Bei der Anzeige wird der Zustand des *lpsf*-Flag wiedergegeben [Kapitel 1.6].
Falls das *dhef*-Flag gesetzt ist, wurde vom System der entsprechende Regelkreis geöffnet. Das Schließen des Regelkreises ist nur nach Systemneustart oder nach Ausführung der Befehle *ra()* oder *rs()* möglich.

4.4 Bitinformationen des ASM-2003 Status-Registers (-> BHB / 4.3.8.5)

Die Zustände der neuen Flags im *asms*-Statusregister werden bei der Statusanzeige in den Feldern {ope}, {options} und {dc} wiedergegeben [Kapitel 1.5].

4.5 Anzeige von Common Variablen (-> BHB / 4.3.8.6)

Es werden jetzt 100 Common Integer- (CI0..CI99) und 100 Common Double-Variablen (CD0..CD99) angezeigt.

5 Inbetriebnahme der Steuerung unter Windows 95 / NT

Für die Inbetriebnahme der Steuerung MCU-3T und MCU-6 unter Windows 95 oder Windows NT gibt es eine TOOLSET Software. Diese muß mit Hilfe des Setup-Programms *setup.exe* auf Diskette1 des „TOOLSET SOFTWARE MCFG for Windows 95/NT"-Diskettensatzes auf dem Zielsystem installiert werden. Als nächster Schritt ist die Registrierung der Hardware notwendig.

5.1 Registrierung der Steuerungshardware

Zur Registrierung der Hardware-Baugruppen muß das ADDIREG–Registrier-Programm in der Programmgruppe A*ddireg* (Voreinstellung) ausgeführt werden.
Dieses Programm startet mit einer Liste von Hardwarebaugruppen. Mit *Insert* kann eine Baugruppe ausgewählt und in die Baugruppen-Liste übernommen werden. Für die MCU-3T muß die Baugruppe PA8000 und für die MCU-6 die Baugruppe PA840 ausgewählt werden. Sofern die vorgeschlagene Basisadresse in Ordnung ist (Interrupts werden i.A. nicht benötigt), kann dies mit „*Set"* und anschließendem „*Save"* bestätigt werden. Die Registrierung sollte mit dem „*Test registration*" Befehl überprüft werden. Beim erstmaligen Registrieren wird gefragt ob der PC neu gebootet werden soll. Dies ist mit „YES" zu bestätigen. Nachdem der PC erneut gebootet wurde, sollte das ADDIREG-Programm erneut aufgerufen und mit dem „*Test registration*"-Befehl die Registrierung überprüft werden. Das Programm kann mit „*Quit*" verlassen werden.
Weitere Informationen zum ADDIREG-Registrierungsprogramm können in dem Dokument ADDIREG.DOC nachgelesen werden. Dieses ist ebenfalls in der Addireg Programmgruppe zu finden.

5.2 Firmware Download

Mit Hilfe des nachfolgend beschriebenen Hilfsprogrammes *mcfg.exe* kann im nächsten Schritt überprüft werden, ob die Steuerung unter Windows 95 oder Windows NT angesprochen werden kann.
Hierzu muß zunächst das Betriebssystem auf die Steuerung geladen werden.

6 Hilfsprogramm mcfg.exe für Windows 95 / NT

Das Hilfsprogramm *mcfg.exe* wurde für die Betriebssysteme Windows 95 und Windows NT in Anlehnung an die DOS-Ausführung erstellt. Mit Hilfe dieses TOOLSET-Programms kann eine schnelle Inbetriebnahme und Projektierung der verschiedenen Steuerungstypen vorgenommen werden. Das Programm unterstützt z.Zt. die Steuerungstypen MCU-3T und MCU-6 mit variabler Achsanzahl.

Beim ersten Start des mcfg.exe-Programms muß eine Systemdatei (normalerweise das File *system.dat*) ausgewählt werden. Dieses File befindet sich je nach Steuerungstyp entweder im gleichnamigen Installations-Unterverzeichnis *sysfile\mcu-3t* oder *sysfile\mcu-6.*
Diese System-Dateien sind voll kompatibel zu den DOS *system.dat*-Files, d.h. bei Bedarf können auch diese Files verwendet werden.
Als nächstes muß die Betriebssystemvariante (*rwtos.btl*) ausgewählt werden. Die Auswahl erfolgt im Menü [File][Project Parameters]. Je nach Steuerungstyp und Achsanzahl kann ein Betriebsprogramm *rwtos.btl* in dem Installations-Unterverzeichnis *rwtos* ausgewählt werden.
Sollte es sich bei der verfügbaren Steuerung um eine vom Standard abweichende Steuerung handeln, ist ein entsprechend angepaßtes *rwtos.btl* Betriebsprogramm notwendig!
Nachdem die korrekten Files (rwtos.btl und system.dat) ausgewählt wurden, kann die Steuerung jetzt mit dem „Boot System“ Befehl im Menü [Tools][System Boot] gebootet werden. Sofern der Bootvorgang erfolgreich war, sollte dies in der Titelleiste des *mcfg.exe* Programms mit der Meldung [Online Mode] bestätigt werden.

Anmerkung: Die System- (system.dat) und Betriebssystem-Dateien (rwtos.btl) befinden sich auf einem separaten Diskettensatz (TOOLSET SOFTWARE WIN32), der wiederum mit einem eigenständigen Installationsprogramm installiert wird.

7 PCAP-Programmierung Windows 95 / NT

7.1 Einführung

Die Steuerungen MCU-3T und MCU-6 können auch unter den Betriebssystem Windows 95 und Windows NT (Win32) eingesetzt werden. Hierzu gibt es eine Win32 TOOLSET-Software die neben dem Hilfsprogramm *mcfg.exe* auch Funktionenbibliotheken für die nachfolgend aufgeführten Programmiersprachen enthält.

Anmerkung: Der Befehlssatz der Win32 Hochsprachen-Bibliotheken ist identisch mit der im [PHB / Kapitel 4.4] beschriebenen PCAP-Befehle, mit der Ausnahme, daß hier bei allen Funktionen der Parameter softint entfällt.

7.1.1 Systemvoraussetzungen

Für alle unten aufgeführten Programmiersprachen gilt: Bevor ein Zugriff auf die Steuerung erfolgen kann muß diese einmalig gebootet werden! Dies kann entweder mit dem Hilfsprogramm *mcfg.exe* oder mit dem *BootFile()*-PCAP-Befehl [Kapitel 1.1] erledigt werden.
Zusätzlich muß folgender Punkt eingehalten werden: Der erste auszuführende PCAP-Befehl muß entweder InitMcuSystem(), InitMcuSystem2() oder InitMcuSystem3() [Kapitel 1.1, 1.3] sein!

7.1.2 Programmiersprache Borland C

Für die Programmiersprache Borland C (C++, C++ Builder) existiert die Datei mcusrvr.h und eine Library pa8000.lib im Installations-Unterverzeichnis BORLAND.

7.1.3 Programmiersprache Borland Delphi

Für die Programmiersprache Borland Delphi (ab Version 2.0) existiert die Datei mcusrvr.pas im Installations-Unterverzeichnis DELPHI.

7.1.4 Programmiersprache Microsoft Visual Basic

Für die Programmiersprache Microsoft Visual Basic existiert die Datei pa8000.bas im Installations-Unterverzeichnis VB.

7.1.5 Programmiersprache Microsoft Visual C++

Für die Programmiersprache Microsoft Visual C++ existiert die Datei mcusrvr.h und eine Library pa8000.lib im Installations-Unterverzeichnis MSVC.

Achtung: Das „Struct member alignment“ muß bei den Compileroptionen auf 4 Byte (Wort-Ausrichtung) eingestellt werden!