POSITIONIER- UND BAHNSTEUERUNG APCI-8001, APCI-8008 und CPCI-8004

Bedienoberfläche einrichten

McuWIN

Stand: 27.01.2015 ab McuWIN V2.5.3.122
Rev. 8/052015

www.addi-data.de

1 Versionsinformation

Die nachfolgende Beschreibung ist gültig für

McuWIN.EXE	ab Version 2.5.3.122
IniCfg.EXE	ab Version 2.5.3.94
MCFG.EXE	ab Version 2.5.3.94
MCUG3.DLL	ab Version 2.5.3.105
NCC.EXE	ab Version 2.5.3.71
RWMOS.ELF	ab Version 2.5.3.123
ToolEdit.exe	ab Version 2.5.3.8

2 Installation

McuWIN ist lauffähig unter fast allen 32-Bit Windows Plattformen (Windows NT, 2000, XP, Vista und Windows 7). Vor der Installation von McuWIN müssen die Programme Miniport, fwsetup und mcfg installiert sein. Die Installation von McuWIN erfolgt durch den Aufruf von SETUP.EXE. Danach wird der Benutzer durch den Installationsvorgang geleitet.
Das Paket kann per Menü oder über die Systemsteuerung wieder deinstalliert werden.

3 Hinweise zum Einrichten

Nach der Installation von McuWIN muss die Umgebung, an die benutzerspezifischen Anforderungen und Gegebenheiten angepasst werden. Die Vorgehensweise hierzu ist nachfolgend beschrieben. Für weitergehende Informationen, Ergänzungswünsche oder Sonstiges stehen wir gerne zur Verfügung:

Telefon: +49 7229 1847-0

E-Mail: info@addi-data.com

3.1 Erstellen eines User-Verzeichnisses

Durch die Installation wurde ein Anwendungsverzeichnis erstellt, welches u.a. die Dateien System.DAT und RWMOS.ELF enthält. Updates und Ergänzungen müssen in dieses Verzeichnis kopiert werden. Das Programm mcfg.exe muss unter „File – Project Parameter“ auf diese Dateien eingestellt werden.
Wenn mcfg vor der Installation von McuWIN installiert wurde, werden die notwendigen Einstellungen automatisch vorgenommen.

3.2 Einrichten der Achsen mit mcfg

Bevor das Programm McuWIN in Betrieb genommen werden kann, muss das komplette System mit dem Einrichtprogramm mcfg.exe eingerichtet und konfiguriert werden. Insbesondere folgende Einstellungen sind wichtig:

- Achsnamen
  Vorsicht: Bei der Verwendung von McuWIN als G-Code Oberfläche dürfen die Achsnamen keine numerischen Zeichen enthalten.
- mechanische Parameter
  (Enkoderauflösung, Getriebefaktor)
- Einstellen der Lageregler
- Maximaler Schleppfehler
- Konfiguration der Ein- und Ausgänge
  Endschalter, Referenzschalter, Verstärkerfreigabe usw.
  siehe hierzu auch Kapitel „Hardware-Voraussetzungen“
- Vorgabe der Jog- und Home- Beschleunigungen und Geschwindigkeiten
- Stop-Deceleration
  (Target-Velocity muss immer 0 sein)
- Deklaration der Software-Endschalter

Die Achsen müssen mit den Motion-Tools in mcfg kontrolliert verfahren werden können. Diese Werte, die in mcfg.exe eingestellt werden, werden beim Speichern in der angegebenen Datei SYSTEM.DAT gespeichert. Wichtig ist, dass McuWIN.EXE ebenfalls auf diese Datei als Systemdatei zugreift.
Vorsicht: falls die Datei SYSTEM.DAT von einer CD herunterkopiert wurde, muss ggf. das Dateiattribut ReadOnly zurückgenommen werden, damit gespeichert werden kann.

3.3 Erster Aufruf von McuWIN

Nun wird das Programm McuWIN zum ersten Mal aufgerufen, und sogleich wieder beendet. Dadurch wird die Datei MCUWIN.INI angelegt oder modifiziert. Diese Datei wird nachfolgend mit dem Programm IniCfg.exe angepasst.

3.4 Einrichten von McuWIN

Die applikationsspezifische Anpassung von McuWIN wird mit dem Programm IniCfg, welches mit McuWIN installiert wurde, durchgeführt. Die durchgeführten Einstellungen werden dabei im Konfigurationsfile McuWIN.INI gespeichert. Manuelle Änderungen im File McuWIN.INI sind nicht mehr erforderlich. Dennoch werden zum Verständnis im Kapitel 5 die Inhalte des INI-Files erläutert.

3.5 Bitvariable anzeigen

Mit dem ebenfalls installierten Programm RegDisp ist es möglich Bitinformationen am Bildschirm anzuzeigen, zu gruppieren und zu beschriften. Bitvariable sind z.B. digitale Eingänge, Ausgänge, Inhalte von CI-Variablen oder aber auch I/Os von zusätzlichen PCI-Baugruppen. Dieses Programm wird besonders während der Systementwicklung hilfreich.
Die Parametrierung der einzelnen Registerkarten und Anzeigeelemente erfolgt per Menüs, die durch Klick mit der rechten Maustaste auf den entsprechenden Anzeigeelementen geöffnet werden.

3.6 Programme editieren

Im Edit-Mode können im Editorfenster von McuWIN Programme geladen, gespeichert und editiert werden. Vor der Programmausführung kann eine Syntaxprüfung durchgeführt werden. Das Ergebnis der Syntaxprüfung wird im Fehlerfenster angezeigt.

3.7 Programme ausführen

Mit dem Start-Button kann das, im Editor aktive Programm, gestartet werden. Der Editor wird dann in den Trace-Modus umgeschaltet.

3.8 Programme schrittweise ausführen

Falls im INI-File entsprechend angewählt, kann mit dem Schritt-Button das Programm im Einzelschrittbetrieb getestet werden. Bei Spooler-Bewegungs-Sequenzen (G01, G02, G03) wird die eigentliche Bewegung erst nach der Ausführung der letzten Zeile, die zum entsprechenden Block gehört ausgeführt. Durch Mausklick in das Editorfenster, kann die Verfahrbewegung vorzeitig gestartet werden.

3.9 Haltepunkte in der Programmausführung

Mit M00 können unbedingte Haltepunkte gesetzt werden. McuWIN geht beim Erreichen von M00 in den Einzelschrittbetrieb über und kann dann per Schritt oder Forts. Button weitergeführt werden.
Mit M01 können bedingte Haltepunkte gesetzt werden. Diese Haltepunkte werden aktiviert durch den Mausbutton „Wahlweiser Halt“ aktiviert werden.

3.10 Teachen von Positionswerten

Mit dem Mausbutton „Teach In Maske“ kann diese geöffnet werden. Hier können die Achsen manuell per Maus-Button verfahren werden. Durch Mausklick auf die Taste Teach wird der aktuelle Positionswert in das Quelltextfile bei der Cursorposition eingefügt.

Das angezeigte Bild im jeweiligen Button kann im File McuWIN.INI in der Section [TEACHMASK] verändert werden (siehe hierzu Abschnitt [TEACHMASK]).

Hinweis: Der Button „Teach“ ist nur bei referenziertem System freigegeben.

Des Weiteren wird durch die Anzeige der Teach-In-Maske das manuelle Verfahren per Joystick freigegeben. Diese Funktionalität ist ab V2.5.3.111 in Task1.SRC enthalten. Weitere Informationen hierzu sind im folgenden Kapitel zu finden.

3.11 Manuelles Verfahren per Analog-Joystick

Zum manuellen Verfahren per Joystick bzw. analogen Eingangssignalen ist die anwenderspezifische Konfiguration einiger Common-Integer-Variablen im Bereich CI300 .. CI399 notwendig. Diese Konfiguration sollte in AppStartInit.inc erfolgen. Um diese Eintragungen zu aktivieren, muss jeweils Task1.src übersetzt und McuWIN.EXE neu gestartet werden, da dieses Include-File in Task1src eingebunden ist.
Von McuWIN.EXE wird in CI300 im Bit 31 angezeigt, ob die Teach-In-Maske aufgelegt ist. Nur in diesem Fall ist das manuelle Verfahren per Analog-Joystick freigegeben. Die Zuordnung der analogen Eingangskanäle zu den physikalischen Achskanälen der Karte ist fest vorgegeben: 1. Achse = Kanal 1, 2. Achse = Kanal 2 usw.
Das Verfahren per Analog-Joystick ist bei den Karten PA 8000 und PS840 nicht möglich.

3.11.1 Initialisierung des Joystickverfahrens in AppStartInit.inc

Zunächst muss das Verfahren per Analogwerte freigeschaltet werden. Dies geschieht in der Variablen CI300. Hier müssen die betreffenden Achsen bitcodiert eingetragen werden. Es ist nur die Verwendung der Achsen 1 bis 8 möglich.

Beispiel:

```
CI300 := $7;      // Achsen 1, 2 und 3 für Joystickverfahren aktivieren
```

Danach müssen die Nullpunkte der einzelnen Analogkanäle in die Variablen CI31x eingetragen werden.

Die eingelesenen Werte der Analogkanäle können in CI39x z.B. mit dem Programm CiShow.exe angezeigt bzw. ermittelt werden, wenn McuWIN aktiv ist und die entsprechenden Kanäle in CI300 freigeschaltet sind. Zur Aktualisierung dieser Werte muss auch die "Teach-In-Maske" in McuWIN geöffnet werden.

Beispiel:

```
CI310 := 16300;
CI311 := 12800;
CI312 := 8635;
```

Nachdem diese Werte in AppTaskInit.inc eingetragen und durch Übersetzen von Task1.src und Neustart von McuWIN aktiviert wurden, kann die jeweilige Vollauslenkung ermittelt und in die Variablen CI32x eingetragen werden.

Beispiel:

```
CI310 := -3200;
CI311 := 2800;
CI312 := 5600;
```

Nach nochmaligem Übersetzen von Task1.src und Neustart von McuWIN steht das manuelle Verfahren per Analog-Joystick zur Verfügung.

3.12 Override

Die Override-Funktion wird im INI-File konfiguriert. Mit dem Schieberegler kann der Override während der Programmausführung eingestellt werden. Der Override-Wert kann auch von einer externen Quelle gesetzt werden. Diese Funktionalität muss vom Anwender z.B. in AppTask2.SRC in einer Endlosschleife programmiert werden. Dies kann über die Systemvariable trovr und den Prozedur-Aufruf utrovr für alle Interpolationsachsen erfolgen.

3.13 Sonstige Hinweise

Nach dem ersten Aufruf von McuWIN sollten insbesondere folgende Werte per IniCfg gesetzt werden:

3.13.1 Verwendete Achsen (usedAxis)

Bitcodierter Zahlenwert der im System verwendeten Achsen:

- Bei diesen Achsen wird der Regelkreis geschlossen
- Für diese Achsen werden Positionsanzeigefenster erzeugt.
- Diese Achsen werden beim Teach-In berücksichtigt
- Diese Achsen können manuell verfahren werden
- Auf diese Achsen wirkt die Override-Funktion
- Erst wenn alle diese Achsen referenziert sind, werden Hardwareendschalter als Fehler in McuWIN angezeigt.

3.13.2 Achsen im Interpolationszusammenhang (InterpolationAxis)

Bitcodierter Zahlenwert der Achsen, die im Interpolationszusammenhang stehen sollen. Hier können u.U. weniger Achsen als in usedAxis angegeben werden.

3.13.3 Kundenspezifische Gestaltung der Programmoberfläche

Ab der Version 2.5.3.24 besteht die Möglichkeit, eine Bitmapdatei in das Zustandsanzeigefenster mit einzubinden. Hierzu muss eine Bitmapdatei mit dem Namen „LOGO.BMP“ in das McuWIN-Arbeitsverzeichnis kopiert werden. Die optimale Auflösung dieser Datei ist 529 x 285 Pixel (H x B) bei einer Bildschirmauflösung von 1024 x 768 Punkten. Auf jeden Fall sollte das Seitenverhältnis eingehalten werden, um Bildverzerrungen zu vermeiden.

3.13.4 Fehler und Warnungen

Die SAP-Programmierumgebung kann Fehlerzustände in der Variablen CI10 bitcodiert setzen. Eine Änderung in diesem Register bewirkt eine Textausgabe im Fehlerfenster der Bedienoberfläche McuWIN (Kapitel 5.7). Somit sind 32 unterschiedliche Fehleranzeigen möglich. Ab McuWIN V2.5.3.102 wurde die Fehleranzeigemöglichkeit ergänzt um die Common Variable CI48. In CI48 können ebenso Fehlerbits gesetzt werden wie bisher in CI10. Die entsprechenden Fehlermeldungen sind im Ini-File in der Section [FEHLERTEXTE48] vorzunehmen, analog wie für Fehleranzeigen zu CI10.
Zusätzlich zu den Fehlerzuständen können auch Warnungen bzw. Statusinformationen angezeigt werden. Hierzu wird die Variable CI15 bitcodiert beschrieben (Kapitel 5.8). Mit dieser Methode sind maximal 31 verschiedene Warntexte möglich. Das Bit 31 (8000 0000 hex) in CI15 ist reserviert zum Löschen der Anzeige im Fehlerfenster.
Beim Setzen oder Rücksetzen von Bits in diesen Registern ist darauf zu achten, dass nur die betreffenden Bits per logische Verknüpfung zu beeinflussen sind:
Setzen von Bits durch Oder-Verknüfpung (OR)
Rücksetzen von Bits durch Und-Verknüpfung (AND)

4 Hardware-Voraussetzungen

4.1 Hauptspindel

Die Hauptspindel wird mit einem Digital-Ausgang Ein- oder Ausgeschaltet. Die Richtung der Hauptspindel wird ebenfalls mit einem Digital-Ausgang geschaltet. Die Belegung dieser Ausgänge muss mit den programmierten Daten in TASK0.SRC bzw. den entsprechenden Include-Files übereinstimmen.
Seit McuWIN Version 2.5.3.59 ist die Deklaration einer Spindelachse in IniCfg.EXE möglich. Falls hier eine Achse definiert ist, ist die hier beschriebene Hauptspindelbehandlung nicht aktiv.

4.2 Ausgang Programm-Ende

Beim Start eines Automatik-Programms wird von McuWIN in Task 0 der Sonderbefehl 1001 aufgerufen. Damit wird der Ausgang Programm-Laeuft gesetzt. Nach der Beendigung eines Automatik-Programms wird der Ausgang Programm-Laeuft zurückgesetzt. Dies erfolgt durch den Aufruf des Sonderbefehls 1000 von McuWIN. Der Ausgang Programm- Laeuft ist standardmäßig O8 des ersten Achskanals.

Beim ersten Einzelschritt eines Programms wird von McuWIN der Sonderbefehl 1002 in Task 0 aufgerufen. In diesem wird ebenfalls der Ausgang Programm-Laeuft gesetzt.

4.3 Ausgang Kühlung

Die Belegung des Ausgangs Kühlung muss mit den programmierten Daten in TASK0.SRC bzw. den entsprechenden Include-Files übereinstimmen.

4.4 Übersicht Ausgangsbelegung

Tabelle: Belegung der Ausgänge der APCI-8001

Achse	Eingang	Pin / Karte / Stecker
1	O1	
1	O2	
1	O3	
1	O4	Kühlung Ein
1	O5	
1	O6	Richtung Hauptspindel
1	O7	Hauptspindel Ein
1	O8	Anzeige Programm Ende

Die Deklarationen der Ausgänge befinden sich in GCode.INC und können bei Bedarf angepasst werden. Falls die entsprechenden G- und M-Befehle nicht oder in anderer Form verwendet werden sollen, müssen die entsprechenden Befehle in den Include-Files redeklariert werden.

4.5 Referenzschalter

Falls die Achsreferenzierung auf Referenzschalter erfolgen soll, sollten die den Achsen entsprechenden Latcheingänge als Referenzschaltereingänge verwendet werden.

Tabelle: Zuordnung Achse / Latcheingang

Achse	Eingang	Pin / Karte / Stecker
1	I14	47 / APCI-8001 / X1
2	I15	48 / APCI-8001 / X1
3	I16	49 / APCI-8001 / X1
4	I30	47 / OPMF / X1
5	I31	48 / OPMF / X1
6	I32	49 / OPMF / X1
7	I39	15 / OPMF / X2 (SUB-D ext.)
8	I40	16 / OPMF / X2 (SUB-D ext.)

5 Variable in McuWIN.INI

Die nachfolgenden Einstellungen werden durch das Programm IniCfg.exe durchgeführt und von McuWIN.exe zur Konfiguration und Anpassung an die Applikation verwendet. Ein manuelles Editieren dieser Datei ist im Allgemeinen nicht erforderlich, da die nachfolgenden Einstellungen in IniCfg.exe editiert werden. Dieser Abschnitt kann jedoch herangezogen werden, wenn der Sinn der einzelnen Einstellungen näher erläutert werden soll.

5.1 [DESKTOP]

In diesem Abschnitt wird das Erscheinen der Programmoberfläche verwaltet. Die Werte WindowState, Left, Top, Height, und Width speichern die Lage der Programmoberfläche und werden von McuWIN verwaltet. Weitere Parameter werden nachfolgend beschrieben.

5.1.1 Single-Step

Merker zum Ein- / Ausblenden des Einzelschritt-Buttons (Standard: 1 = Ein).

5.1.2 Override

Merker zum Ein- / Ausblenden des Override-Trackbar (Standard: 0 = Aus).

5.1.3 OverrideEnable

Merker, ob der Override-Trackbar in der Bedienoberfläche per Maus verändert werden kann (Standard: TRUE). Wenn der Override-Wert z.B. in der Taskumgebung gesetzt werden soll, muss OverrideEnable=0 gesetzt werden. Der Override-Trackbar hat in diesem Fall nur noch Anzeigecharakter. Siehe hierzu auch Kapitel 3.12.

5.1.4 OverrideMax

Maximalwert des Override Trackbar in % (Standard: 125). Dieser Wert wird auch bei einem eventuellen Spindel-Override verwendet.

5.1.5 OverrideMin

Minimalwert des Override Trackbar in % (Standard: 0). Hier sind auch negative Werte möglich. Dann kann innerhalb eines Verfahrprofils auch zurückgefahren werden. Dieser Wert wird auch bei einem eventuellen Spindel-Override verwendet.

5.2 [EDITOR]

5.2.1 SrcFileName

Im Parameter SrcFileName wird der Name und Pfad der User Task gespeichert. Dieser Parameter wird von McuWIN verwaltet.

5.2.2 AutoLoadOnActivate

Wenn dieser Parameter auf 1 gesetzt wird, wird das angegebene Quelltextfile bei jeder Aktivierung von McuWIN unter Windows neu geladen, sobald sich das System im Edit-Mode befindet, d.h. wenn kein Benutzerprogramm ausgeführt wird.

5.2.3 DisableEdit

Wenn dieser Parameter auf 1 gesetzt wird, können Benutzerprogramme zwar geladen und ausgeführt, aber nicht editiert werden.

5.2.4 SrcFilter

Hier kann eine Standard-Extension für Quelltextfiles angegeben werden. Diese Extension wird nach erstmaliger Anwahl beim Öffnen einer Datei immer wieder vorgeschlagen, wenn eine Quelltextdatei geöffnet werden soll.

5.3 [MCU]

In dieser Section befinden sich Steuerungsspezifische Parameter.

5.3.1 AutoSetNum

Wenn dieser Merker gesetzt ist (≠ 0), dann werden im G-Code Modus die sonst erforderlichen Zeilen-Nummern (Nxxx) nicht benötigt.

5.3.2 usedAxis

Angabe der tatsächlich verwendeten Achsen (bitcodiert). Mit dieser Variablen könne die Achsen des Betriebssystems, die von McuWIN bearbeitet werden, eingeschränkt werden. (Standard FFh)
Beispiel: Sie haben ein System mit 3 Achsen (X, Y und Z). Dann muss hier der Wert 7 eingetragen werden, weil hier die drei niederwertigsten Bits gesetzt sind.

5.3.3 InterpolationAxis

Angabe der Achsen, die im Interpolationszusammenhang stehen!

5.3.4 EECheck

Angabe der Achsen (bitcodiert), bei denen der Enkoder-Error überprüft wird. Bei Schrittmotorsystemen kann diese Variable auf 0 gestellt werden. Mit dieser Variable wird gleichzeitig auch die Positions-Überwachung per Index-Latch-Funktion aktiviert.

5.3.5 FehlerVarCI, RefVarCI

Angabe der CI-Variable für Fehleranzeige und Referenzzustandsanzeige. Diese Variable sind durch die Programme TASK0.SRC und TASK1.SRC vorgegeben und dürfen nicht verändert werden.

5.3.6 RefAfterEO

Mit diesem Merker wird angegeben, ob das System nach einem Not-Aus-Ereignis neu referenziert werden muss (Standardwert=1).

5.3.7 BaseAdress

Bei der PA8000 und der PS840 muss hier die Basisadresse der Steuerung eingetragen werden (Standard = 768 = 300H).

5.3.8 SystemFileName

Pfad und Name des Systemfiles (Standard: SYSTEM.DAT im Programmverzeichnis).

5.3.9 BootFileName

Pfad und Name des Bootfiles (Standard: RWMOS.ELF im Programmverzeichnis).
Vorsicht: Bei der PA8000 und der PS840 muss hier i.A. RWTOS.BTL eingetragen werden.

5.3.10 Task?Aktiv

? ist der Index der Task 0..3. Mit dem jeweiligen Merker wird angegeben ob die jeweilige Task beim Programmstart geladen werden soll. Falls ja wird unter ***Task?CncFileName*** der Name des zu ladenden Files mit Pfad angegeben (Standard: TASK?.CNC im Programmverzeichnis).
Im Zusammenhang mit Task 3 können zwei weitere Dateinamen angegeben werden: ***PreIncFileName*** gibt den Namen eines Include-Files an, welches am Anfang des G-Code-Files eingebunden wird. Diese Include-Datei kann Variablen- und Prozedur-Deklarationen in rw_SymPas-Syntax enthalten, die im Programm verwendet werden können, die aber für den Maschinenbediener nicht sichtbar sind.
PostIncFileName gibt den Namen eines Include-Files an, welches am Ende des G-Code-Files eingebunden wird. Diese Include-Datei kann Deklarationen von Unterprogrammen in G-Code-Syntax enthalten, die im Programm verwendet werden können, die aber für den Maschinenbediener nicht sichtbar sind.

5.3.11 TRAC, TRVL

Hier werden die Standardwerte für Bahngeschwindigkeit und Bahnbeschleunigung für Interpolationsbefehle eingetragen. Die Einheiten für diese Werte sollten in der Überwachungstask gesetzt werden. Falls diese dort nicht gesetzt werden sind mm/sec und mm/sec^2 angewählt.

5.3.12 FreeingVelFactor

Freifahrbewegungen z.B. bei der Referenzierung von Achsen, werden normalerweise mit der Eilgang-Geschwindigkeit / -Beschleunigung (Jog-Velocity, Jog-Acceleration) durchgeführt. Mit diesem Parameter, können diese Werte vermindert oder erhöht werden. Dieser Faktor wirkt für alle Achsen.

5.3.13 ReferenceSwitchLatch

In diesem Merker sind bitcodiert die Achsen angegeben, bei denen der Referenzschaltereingang gleichzeitig ein schneller Latcheingang ist. Wenn dies nicht der Fall ist, oder wenn die Latchsignalerkennung nicht überprüft werden soll, kann dies mit Hilfe dieses Merkers achsspezifisch deaktiviert werden.

5.3.14 NoRefToLimitSwitch

Wenn nicht auf einen Referenzschalter referenziert wird, erfolgt die Referenzierung normalerweise auf einen Endschalter. Wenn dies auch nicht gewünscht ist, kann dies mit Hilfe dieses Merkers (bitcodiert) deaktiviert werden. In diesem Fall wird nur auf das Indexsignal oder überhaupt nicht referenziert.

5.3.15 ReferenzUeberwachung

In diesem Parameter wird bitcodiert angegeben welche Achsen referenziert sein müssen, damit ein Programm abgefahren werden kann (Standard 0xFF);
Als Referenzschalter müssen jeweils die Latcheingänge der einzelnen Achsen (Eingänge 14, 15 und 16 bei den ersten drei Achsen) verwendet werden.

5.3.16 UeberwachungsTask

Hier muss die Task angegeben werden, die das System überwacht. Wenn keine Überwachungstask vorhanden ist, muss hier ein negativer Wert eingetragen werden (Standard: -1).
Die Überwachungstask muss auch mit Task?Aktiv=1 aktiviert und mit Task?Autorun=1 automatisch gestartet werden.
Mit Task?CncFileName= wird der Dateiname mit Pfad angegeben (Standard: TASK?.CNC).

5.3.17 UserTask

Hier muss die Task eingetragen werden, in der das Programm im Editor ausgeführt werden soll. Standard ist 0.

5.3.18 DINGCodes

Mit dem Wert 1 für diesen Parameter kann die Option G-Code Verarbeitung aktiviert werden. Die UserTask muss dann Task 3 sein. Hierzu muss eine Befehlsinterpretertask in Task 0 und eine Überwachungstask in Task 1 geladen werden.

5.3.19 UserTaskAutoRun

Mit diesem Merker wird angegeben ob die UserTask automatisch nach dem Programmstart gestartet werden soll.

5.3.20 UserTaskAutoStop

Mit diesem Merker wird angegeben, ob das System nach beenden von McuWIN rückgesetzt werden soll. Wenn ja wird das File ***StopUserTask.CNC*** in die User-Task geladen und gestartet.

Vorsicht: ***StopUserTask.CNC*** muss für die richtige Task übersetzt sein.

5.3.21 SProfile

Indikator, gibt an ob S-Profile oder Trapez-Drehzahl-Profile abgefahren werden sollen (Standard = 0).

5.3.22 JerkRel

Faktor für Ruck bei S-Profilen (Standard = 0.5):
0.0 = maximaler Ruck erlaubt, entspricht Trapez Drehzahl Profil
1.0 = minimaler Ruck, entspricht dreieckförmigem Beschleunigungsverlauf
Werte zwischen 0 und 1 entsprechen trapezförmigem Beschleunigungsverlauf

Hier kann ein Gleitpunkt-Wert zwischen 0.0 und 1.0 angegeben werden, wenn SProfile = 1 ist.

5.3.23 LookAhead

Indikator, gibt an ob die Look-Ahead Geschwindigkeitsbegrenzung aktiv ist (Standard = 1). Bei aktivem Look-Ahead wird für die einzelnen Achsen der maximal erlaubte Geschwindigkeitssprung angegeben, der in A?.mdvel angegeben ist.

5.3.24 LookAheadDeep

Ganzzahlvariable, gibt an, bei wie vielen Interpolations-Verfahrbefehlen im Spooler die Spoolerabarbeitung beginnt. Diese Anzahl ist somit auch die maximale Anzahl von Verfahrprofilen, über die eine Look-Ahead-Berechnung durchgeführt werden kann.

5.3.25 NoTriangle

Indikator! Gibt an, ob in Look-Ahead-Profilabschnitten, in welchen die Maximalgeschwindigkeit nicht erreicht wird (Dreiecksprofile) überhaupt hochbeschleunigt werden soll, oder ob diese mit konstanter Geschwindigkeit durchfahren werden sollen. Mit diesem Indikator, kann ein unruhiger. „sägender“ Verfahrvorgang beim Abfahren von kleinen Profilabschnitten mit Look-Ahead-Überwachung verhindert werden. Dadurch wird jedoch die Positionierzeit größer.

5.3.26 A?.mdvel

Achsspezifischer, maximaler Geschwindigkeitssprung im Look-Ahead-Modus (Standard = 0,1). Für das „?“ ist die jeweilige Achsnummer (1..n) maßgeblich. Die Einheit ist die jeweilige Interpolationseinheit.

5.3.27 CenterPointRelative

Merker, mit dem angegeben werden kann ob Kreismittelpunktskoordinaten im Absolutmodus (G90) als Relativ- oder Absolutkoordinaten angegeben werden.

5.3.28 RotatoricUnit

Merker, mit dem angegeben werden kann ob rotatorische Achsen (Achsen mit einer rotatorischen Anwender-Einheit) bei Verwendung in translatorisch definierten Interpolationsfahrten (G01, G02, G03) in der angewählten translatorischen Einheit, oder in der benutzerspezifischen rotatorischen Einheit programmiert werden. Die Umrechung zwischen translatorischen und rotatorischen Größen erfolgt mit dem achsenspezifischen Effektivradius, welcher z.B. mit dem Kommando G51 programmiert werden kann.

5.3.29 ShortestRotatoricDist

Merker, mit dem angegeben werden kann ob rotatorische Achsen (Achsen mit einer rotatorischen Anwender-Einheit) beim absoluten Verfahren jeweils die kürzeste Distanz zum Anfahren des Zielpunktes verwenden. Wenn dieser Merker gesetzt ist, würde eine Achse bei einem Verfahrweg von 359 Grad nach 1 Grad um 2 Grad in positive Richtung verfahren. Wenn der Merker nicht gesetzt ist, verfährt die Achse um 358 Grad in negative Richtung.

5.3.30 CmdTpDisplay

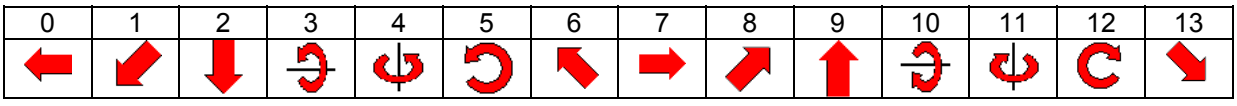
In der Oberfläche wird als Sollposition die Desired Position der jeweiligen Achsen angezeigt. Wenn der Merker CmdTpDisplay gesetzt ist, wird als Sollposition die Zielposition des aktuellen Profils angezeigt.

5.4 [TEACHMASK]

In dieser Section kann jedem Verfahrbutton eine Bitmap zugeordnet werden.

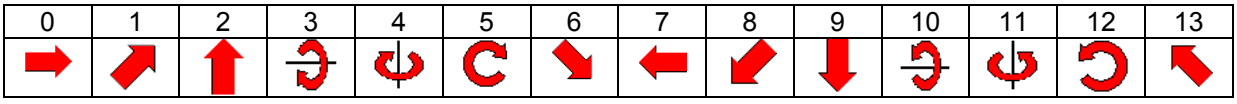
5.4.1 GlyphLeftX=Y

Mit dieser Anweisung wird den Buttons der linken Spalte ein Bild zugeordnet. X ist die Zeile des Buttons (Zählweise von 0 beginnend). Y ist der Index der verfügbaren Buttons laut Tabelle.



5.4.2 GlyphRightX=Y

Mit dieser Anweisung wird den Buttons der rechten Spalte ein Bild zugeordnet. X ist die Zeile des Buttons (Zählweise von 0 beginnend). Y ist der Index der verfügbaren Buttons laut Tabelle.



5.5 [SYSTEM]

5.5.1 CompileAlways

Wenn dieser Merker <> 0 ist, wird das aktive SRC-File bei jedem Programmstart zunächst übersetzt und dann auf die Steuerung geladen. Wenn CompileAlways = 0 gesetzt ist, wird zunächst überprüft, ob das CNC-File bereits existiert. Wenn ja, dann wird das SRC-File nur übersetzt, wenn das Datum des SRC-Files neuer ist, als das Datum des CNC-Files.
Somit kann in vielen Fällen die Zeit für den (unnötigen) Übersetzungsvorgang eingespart werden. Dies ist insbesondere interessant bei langsamen PCs oder bei sehr großen Programmen.

5.5.2 RebootEnabled

Mit diesem Merker kann gesteuert werden, ob das System durch einen Doppelklick auf das Fehlerfenster neu gebootet wird.

5.5.3 Achskompensation

Mit diesem Merker kann eine Achskompensation geladen werden. Hierzu wird auf ein gesondertes INI-File zugegriffen, dessen Name und Pfad in *KompensationsFile* angegeben wird (Standard: CP.INI).
Dieses INI-File kann mit dem Programm WrParams.EXE erstellt werden. Falls diese Option verwendet wird, muss RWMOS mit der Option GEAR verwendet werden.
Vorsicht: Im Ini-File muss der Pfad angegeben sein, sonst wird die Datei nicht gefunden!

5.5.4 HWStart(n)

Mit dem Merker HWStart können ein oder mehrere Digitaleingänge bitweise angegeben werden, mit denen ein Hardware-Programm-Start erfolgen kann. n ist die Nummer des Achskanals. Diese Funktion ist nur wirksam wenn im Editor keine ungespeicherten Änderungen vorliegen.
Die Anwahl von Bit 16 bewirkt, dass Bit 0 von CI63 (1 hex) zum Programmstart herangezogen wird.

Beispiel: Der Eingang 2 der X-Achse (erste Achse) soll einen Start des Anwenderprogramms bewirken

HWStart1=2

5.5.5 HWStop(n)

Mit dem Merker HWStop können ein oder mehrere Digitaleingänge bitweise angegeben werden, mit denen ein Hardware-Programm-Stop erfolgen kann. n ist die Nummer des Achskanals.
Die Anwahl von Bit 16 bewirkt, dass Bit 3 von CI63 (8 hex) zum Programmstart herangezogen wird.

Beispiel: Der Eingang 3 der X-Achse (erste Achse) soll einen Stop des Anwenderprogramms bewirken

HWStop1=4

5.5.6 HWRef(n)

Mit dem Merker HWRef können ein oder mehrere Digitaleingänge bitweise angegeben werden, mit denen ein Hardware-Start der Referenzfahrt erfolgen kann. n ist die Nummer des Achskanals.
Die Anwahl von Bit 16 bewirkt, dass Bit 2 von CI63 (4 hex) zum Programmstart herangezogen wird.

Beispiel: Der Eingang 3 der X-Achse (erste Achse) soll die Referenzfahrt starten

HWRef1=4

5.5.7 HWReset(n)

Mit dem Merker HWReset können ein oder mehrere Digitaleingänge bitweise angegeben werden, mit denen eine Hardware-Betätigung des Reset-Buttons erfolgen kann. n ist die Nummer des Achskanals.
Die Anwahl von Bit 16 bewirkt, dass Bit 1 von CI63 (2 hex) zum Programmstart herangezogen wird.

Beispiel: Der Eingang 4 der X-Achse (erste Achse) soll einen Reset bewirken

HWReset1=8

5.5.8 HWSingleStep(n)

Mit dem Merker HWSingleStep können ein oder mehrere Digitaleingänge bitweise angegeben werden, mit denen eine Hardware-Betätigung des SingleStep-Buttons erfolgen kann. n ist die Nummer des Achskanals.
Die Anwahl von Bit 16 bewirkt, dass Bit 6 von CI63 (40 hex) zum Programmstart herangezogen wird.

Beispiel: Der Eingang 5 der X-Achse (erste Achse) soll einen Reset bewirken

HWSingleStep1=$10

5.5.9 HilfsSpannungEin(n)

Inicfg.exe-Seite „Hardware Ausgänge“, Zeile „Hilfsspannung Ein:“

Mit dem Merker HilfsSpannungEin kann ein Digitalausgang angegeben werden, mit dem ein Hardware-System-Start erfolgen kann. n ist die Nummer des Achskanals (Zählweise von 1 beginnend). Wenn hier ein Ausgang angegeben ist, wird in McuWIN nach dem Start oder nachdem alle Fehler in CI10 aufgehoben sind eine Bildschirmmaske angezeigt. In dieser Maske kann über eine Schaltfläche der angegebene Digitalausgang gesetzt werden. Sobald in CI10 ein Fehler angezeigt wird, wird dieser Ausgang wieder zurückgenommen.

Beispiel: Der Ausgang 2 der X-Achse (erste Achse) soll für das Einschalten der Hilfsspannung verwendet werden.
HilfsSpannungEin1=2

5.5.10 UserSpecCode

Hier kann ein benutzerspezifischer Wert eingegeben werden, um in McuWIN applikationsspezifische Eigenschaften freizuschalten.

5.6 [REFERENZFAHRT]

5.6.1 Reihenfolge

Reihenfolge der Referenzierung. Die Achsnummern werden in den einzelnen Dezimalstellen abgelegt (Zählweise mit 1 beginnend). Die Achse in der niederwertigsten Stelle wird zuerst referenziert (Einerstellen).

Beispiel:
Reihenfolge=123

Hinweis: Bei der Inbetriebnahme geht man am besten derart vor, dass man hier zunächst nur die Achse angibt, die zuerst referenziert werden soll, und mit dieser eine erfolgreiche Referenzfahrt durchführt (z.B. Z-Achse = 3. Achse)
Reihenfolge=3

Danach hängt man die nächste Achse vor diese Ziffer (z.B. X-Achse = 1. Achse).
Reihenfolge=13

Dies setzt man fort, bis alle Achsen erfolgreich referenziert werden.

5.6.2 ReferenzSchalter

Angabe der Achsen, bei denen auf Referenzschalter referenziert wird (bitcodiert).

5.6.3 Indexsuche=0

Angabe der Achsen, bei denen auf den Indeximpuls referenziert wird (bitcodiert).

5.6.4 Richtung

Referenz-Suchrichtung (bitcodiert). 1 bewirkt Referenzierung in negative Richtung.

5.6.5 GotoZeroAfterRef

Mit dieser Variable wird bitcodiert angegeben, ob die entsprechende Achse nach der Referenzierung ihren Nullpunkt anfahren soll. Der Standardwert ist 0.

5.6.6 ReferenzPosX / Y / ...

Referenzposition im Referenzpunkt.

5.6.7 RuhePosX / Y / ...

Ruheposition, kann nach Referenzierung angefahren werden.

5.7 [FEHLERTEXTE]

In diesem Abschnitt werden Anzeigetexte für den Fehlerfall definiert. Einige Fehler sind von McuWIN bereits vordefiniert. Diesen Abschnitt kann man zunächst aus der Datei „FehlerTexte.INI“ kopieren. Weitere Fehler können applikationsspezifisch ergänzt werden. Der Aufruf der Fehlertexte erfolgt bitcodiert über die Variable CI10.
Ab McuWIN V2.5.3.103 steht als Fehlervariable, insbesondere für anwenderspezifische Erweiterungen, die Variable CI48 und die zugeordnete Section [FEHLERTEXTE48] zur Verfügung.

5.7.1 Benutzerspezifische Fehlertexte

Section [FEHLERTEXTE] Abschnitt Fehlernummer (dezimal)
Hier muss ein Text eingetragen werden. Dieser Text muss %X für die Fehlernummer enthalten.

5.7.2 [FEHLERINFO]

Abschnitt Variablennummer (dezimal)
Hier muss eine CI-Adresse eingetragen werden, welche eine oder zwei Achsen spezifiziert, die den Fehler ausgelöst hat. Wenn Zwei Achsen anzugeben sind, muss die linke Achse * 100 angegeben werden (z.B. 1213 – Achse links in CI12 und Achse rechts in CI13)
Für jede Achse die spezifiziert wird, wird automatisch ein Fehlerstring angehängt in dem der Achsname angezeigt wird.
Beispiel: es sind zwei Achsen ausgewählt

```
[FEHLERTEXTE]
1=Fehler # %i: Unbekannter Funktionscode!
2=Fehler # %i: Schleppfehler
[FEHLERINFO]
1=0
2=14
```

5.7.2.1 Fehlerliste

5.7.2.1.1 200 hex (512 dez) Fehler bei Werkzeug-Radius-Korrektur

Bei diesem Fehler wurde ein Fehler im TC-Modul des universellen Objekt-Interface erkannt. Entweder der Zugriff auf eine Funktion war nicht möglich, oder die Modulvariable "ERROR" lieferte ein oder mehrere Fehlerbits zurück. Der Fehler des TC-Moduls wird als Fehlercode ebenfalls hexadezimal angezeigt. Die Bedeutung dieser Fehler sind im Handbuch "TC-Interface" in Tabelle 4 aufgelistet. Hierbei ist eine Kombination mehrerer Fehlerbits möglich.

5.8 [WARNTEXTE]

In diesem Abschnitt sind Warn- oder Status-Texte definiert. Diese sind bitcodiert und werden ausgegeben, wenn in CI15 das entsprechende Bit gesetzt ist. Mit der Text-Ausgabe wird das entsprechende Bit in CI15 zurückgenommen. Wenn zusätzlich das höchstwertige Bit in CI15 gesetzt ist, wird vor der Ausgabe das Fehlerfenster gelöscht. Warntexte werden nur ausgegeben, wenn kein Fehler (Bit in CI10) aktiv ist.

5.8.1 Benutzerspezifische Warntexte

Section [FEHLERTEXTE] Abschnitt Fehlernummer (dezimal)
Hier muss ein Text eingetragen werden. Dieser Text muss %X für die Warnung- / Status-Nummer enthalten.

5.9 [TOOLCOMPENSATION]

5.9.1 TcFileName

Mit dieser Variable kann ein Ini-File mit Informationen über die Werkzeugradiuskorrektur angegeben werden. Musterdaten hierzu liegen in „ToolComp Demo.ini“ vor.

5.10 [DDE]

McuWIN bietet die Möglichkeit mit einem DDE-Master zu kommunizieren. Hierzu ist Abschnitt „DDE Kommunikation“ zu beachten. Die Einstellungen für die DDE-Kommunikation erfolgen in der Section DDE.

5.11 [SPINDEL]

In diesem Abschnitt kann die Funktion einer Spindelachse konfiguriert werden. Falls eine Spindelachse konfiguriert wurde, wird die Behandlung der Hauptspindel nach Kapitel 4.1 unwirksam.
In dieser Funktionalität ist ein Spindel-Override enthalten, welcher es erlaubt die Spindeldrehzahl manuell nachzujustieren. Beim Gewindeschneiden sind Spindel- und Achs-Override synchronisiert. Falls der Spindel-Override anders als mit dem Override-Slider eingestellt werden soll, muss hierzu der JOG-Override der Spindelachse (z.B. A.jovr) beschrieben werden. Der Wert 1 beim Jog-Override entspricht 100%.

5.11.1 SpindelAxis

Angabe eines Achskanals, welcher als Spindelachse fungieren soll. Bei dem Wert 0, welcher der Eingabe OFF im Programm IniCfg entspricht, ist die Behandlung einer Spindelachse deaktiviert.
Dieser Wert wird per CI64 in den Bits 0..3 an RWMOS.ELF übergeben.

5.11.2 SpindelType

In dieser Variablen wird angegeben, um welchen Spindeltyp es sich handelt.

Wert	Spindeltyp
0	Analog-Ausgang
1	geregelte Achse
2	Anwenderprogrammiert

Dieser Information wird per CI64 in den Bits 4..7 an RWMOS.ELF übergeben, jedoch nicht als der o.g. Zahlenwert sondern in bitcodierter Form.

5.11.3 InSpindelReady

In dieser Variablen kann die Nummer eines Eingangs der Spindelachse definiert werden, welcher anzeigt, daß die Spindeldrehzahl erreicht wurde. Mit dem Wert 0 wird die Funktion deaktiviert. Beim Einschalten der Spindel verweilt die Programmausführung im entsprechenden Befehl (z.B. M03), bis die Spindeldrehzahl erreicht ist.
Dieser Wert wird per CI64 in den Bits 8..15 an RWMOS.ELF übergeben.

5.11.4 SpindelOvrVisible

Mit dieser Variablen, kann angegeben werden ob der Override-Slider für den Spindel-Override in der Programmieroberfläche sichtbar ist. Mit dem Wert 1 wird der Slider eingeblendet. Dieser Wert wird per CI64 in Bits 16 an RWMOS.ELF übergeben. Die Maximal- und Minimalwerte des Spindel-Override-Sliders werden an die Einstellungen des Verfahr-Override-Sliders angepasst.

5.11.5 SpindelOvrEnabled

Mit dieser Variablen wird definiert, ob der Spindel-Override-Slider mit Tastatur und Maus bedienbar ist.
Dieser Wert wird per CI64 in Bit 17 an RWMOS.ELF übergeben.
Eine Alternative zur Bedienung des Sliders per Tastatur/Maus ist z.B. die Auswertung eines Analogwertes von einem Potentiometer. Die entsprechende Auswertung muss vom Anwender implementiert werden z.B. in AppTask2.src. Die Übergabe des Override-Wertes an die Steuerung erfolgt über den JOG-Override der Spindelachse.
Dieser Wert wird per CI64 in Bits 17 an RWMOS.ELF übergeben.

5.11.6 SpindelMaxVelocity

In dieser Variablen wird die Maximaldrehzahl der Spindel in der Anwendereinheit (z.B. U/min) definiert.
Dieser Wert wird per CI65 an RWMOS.ELF übergeben.

5.11.7 SpindelSollIst

In dieser Variablen wird die Information, ob eine Soll-/Istwertanzeige der Spindelachse in der Oberfläche erfolgen soll, übergeben. Wenn die Soll-/Istwertanzeige aktiviert ist, wird auch der symbolische Achsname der Spindelachse angezeigt.

6 Variablenbelegung

Zur Bedeutung der Common-Variablen ist unbedingt das Kapitel 10.2 zu beachten.

6.1 CD-Variable

Nr.	Bedeutung	Programm
0-17	Parameter von G, T oder anderen Codes Die verwendeten Variable sind in der Systemvariablen AXSEL bitweise codiert.	McuWIN, TASK0
20	TRAC – Standard Bahnbeschleunigung	McuWIN, TASK1
21	TRVL – Standard Bahngeschwindigkeit	McuWIN, TASK1
23	Zwischengespeicherter Wert von Override bei Vorschubsperre	TASK0, TASK1
24	Geschwindigkeitsfaktor für Freifahren	TASK0
30	Start-Positionswert A1 für Positionsüberwachung	TASK1
31	Start-Positionswert A2 für Positionsüberwachung	TASK1
32	Start-Positionswert A3 für Positionsüberwachung	TASK1
33	Start-Positionswert A4 für Positionsüberwachung	TASK1
34	Start-Positionswert A5 für Positionsüberwachung	TASK1
35	Start-Positionswert A6 für Positionsüberwachung	TASK1
36	Start-Positionswert A7 für Positionsüberwachung	TASK1
37	Start-Positionswert A8 für Positionsüberwachung	TASK1
CD40	Wert max. Geschwindigkeitssprung {mdvel} für Achse A1	TASK1, McuWIN
CD41	Wert max. Geschwindigkeitssprung {mdvel} für Achse A2	TASK1, McuWIN
CD42	Wert max. Geschwindigkeitssprung {mdvel} für Achse A3	TASK1, McuWIN
CD43	Wert max. Geschwindigkeitssprung {mdvel} für Achse A4	TASK1, McuWIN
CD44	Wert max. Geschwindigkeitssprung {mdvel} für Achse A5	TASK1, McuWIN
CD45	Wert max. Geschwindigkeitssprung {mdvel} für Achse A6	TASK1, McuWIN
CD46	Wert max. Geschwindigkeitssprung {mdvel} für Achse A7	TASK1, McuWIN
CD47	Wert max. Geschwindigkeitssprung {mdvel} für Achse A8	TASK1, McuWIN
CD48	JerkRel	TASK1, McuWIN
CD50	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A1	TASK0
CD51	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A2	TASK0
CD52	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A3	TASK0
CD53	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A4	TASK0
CD54	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A5	TASK0
CD55	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A6	TASK0
CD56	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A7	TASK0
CD57	Rückgabe akt. gesetzte Nullpunktverschiebung Achse A8	TASK0
CD60	Referenzposition Achse A1	TASK0, McuWIN
CD61	Referenzposition Achse A2	TASK0, McuWIN
CD62	Referenzposition Achse A3	TASK0, McuWIN
CD63	Referenzposition Achse A4	TASK0, McuWIN
CD64	Referenzposition Achse A5	TASK0, McuWIN
CD65	Referenzposition Achse A6	TASK0, McuWIN
CD66	Referenzposition Achse A7	TASK0, McuWIN
CD67	Referenzposition Achse A8	TASK0, McuWIN
CD68	Stellwert für Analogwert-Spindelausgang	TASK0, TASK1

Nr.	Bedeutung	Programm
CD70	Ruheposition Achse A1	TASK0, McuWIN
CD71	Ruheposition Achse A2	TASK0, McuWIN
CD72	Ruheposition Achse A3	TASK0, McuWIN
CD73	Ruheposition Achse A4	TASK0, McuWIN
CD74	Ruheposition Achse A5	TASK0, McuWIN
CD75	Ruheposition Achse A6	TASK0, McuWIN
CD76	Ruheposition Achse A7	TASK0, McuWIN
CD77	Ruheposition Achse A8	TASK0, McuWIN
CD80	Verfahrweg für manuelles Verfahren (0 = Endlos) in UU	Task1
90	Version von TASK0.SRC	TASK0
91	Version von TASK1.SRC	TASK1
92	Version von TASK2.SRC	TASK2
94	Version von McuWIN.EXE	McuWIN
95	Version von RWMOS.ELF	TASK1
600 bis 699	Applikationsspezifisch	

6.2 CI-Variable

Nr.	Bedeutung	Programm	
0	Befehlsübergabe für Befehlsinterpreter G	McuWIN.EXE	
1	Befehlsübergabe für Befehlsinterpreter M	McuWIN.EXE	
2	Befehlsübergabe für Befehlsinterpreter Sonderbefehle	McuWIN.EXE	
3	Kommando von Task -> McuWIN.EXE 1 = Fehlerfenster löschen und andere	Task0 / McuWIN	
4	Parameter von Task -> McuWIN.EXE	Task0 / McuWIN	
9	Achsen im System bitcodiert (reduziert)	McuWIN.EXE T0	
10	Fehlervariable Diese Variable ist bitcodiert. Die einzelnen Fehler sind in GCODE.INC definiert.	McuWIN.EXE T0	
11	Achsen referenziert (bitcodiert)	McuWIN.EXE T0 T1	
12	Merker: Protokolldatensatz aufzeichnen	T0 T1	
13	Achsen für EE-Überwachung (bitcodiert in bit 0..15) und für Zählerüberwachung per Index (bit 16 .. 23)	McuWIN.EXE T1	
14	(WW: MPE verursachende Achse) Merker: Task3 nicht starten nach Bef.Ausführung	(McuWIN.EXE) McuWIN.EXE T0	
15	Warnung Nr. bitcodiert MSB (Bit 31) gesetzt heißt: Fehlerfenster löschen. Dieses Bit wird nach Setzen aus einer Task von McuWIN.exe abgelöscht!	MCUWIN.EXE T1	
16	Anzahl aller tatsächlich vorhandenen Achsen nach Boot der Steuerung	McuWIN.EXE T1	

Nr.	Bedeutung	Programm	
17	Achse Nr. (bitcodiert) bei: Achse bei Fehler Referenzfahrt Achse bei Fehler MPE Achse bei Konfigurationsfehler Achse bei Enkoder-Error Achse bei Zähler-Fehler (erkannt per Index)	T0 T1 McuWIN.EXE	
18	Achse bei ESL Hard + Soft (bitcodiert)	T1	
19	Achse bei ESR Hard + Soft (bitcodiert)	T1	
20	Status der Zählerüberwachung (bitcodiert)	T1	
Nr.	Bedeutung	Programm	
21	Reihenfolge der Referenzfahrt (Dezimalstellen Codiert)	McuWIN.EXE T0	
22	Referenzierung auf Referenzschalter? (bitcodiert)	McuWIN.EXE T0	
23	Indexsuche bei Referenzfahrt? (bitcodiert)	McuWIN.EXE T0	
24	Richtung der Referenzfahrt (bitcodiert)	McuWIN.EXE T0	
25	Index der X-Achse (für ToolCompensation)	T1 McuWIN.EXE	
26	Index der Y-Achse (für ToolCompensation)	T1 McuWIN.EXE	
27	Index der Z-Achse (für ToolCompensation)	T1 McuWIN.EXE	
28	Kundenspezifische Referenzfahrt durchführen, wenn <> 0	T1 McuWIN.EXE	
29	Nullpunkt nach Ref.Fahrt anfahren? (bitcodiert)	T1 McuWIN.EXE	
30	Zeigt bitcodiert Optionen von RWMOS.ELF an Bit 0 = Ressourcen-Interface vorhanden Bit 1 = TC-Interface vorhanden Bit 2 = TC-Tabellenwerte sind programmiert		
31	Statusinfo für Wahlweiser oder unbedingter Halt	McuWIN.EXE	
32	TC-Error Wort	TASK1.SRC	
33	Konfigurationswort für MODEREG - S-Profil - Look-Ahead		
34	Steuerungsgeneration PA xxxx und PSxxx = 2, xPCI-800x = 3	McuWIN.EXE alle Tasks je nach Bedarf	
35	LookAhead-Tiefe	McuWIN.EXE	
36	Merker, ob System nach Not-Aus dereferenziert ist	McuWIN.EXE TASK1.SRC	
37	Info ob Programm läuft: 0 – Programm steht 1 – Programm läuft 2 – Programm ist aktiv im Einzelschritt-Modus	McuWIN.EXE	
38	zusätzliche Fehlerzustandsinfo	T0, T1	
39	Bits, um Achsen manuell zu verfahren	T2 McuWIN.EXE	
40ff	Oelmaier CNC-7A: Ausgangsvariable für AWS		
48	Applikationsspezifische Erweiterung zur Fehlervariablen CI10 (bitcodiert). Siehe hierzu auch Kap. 3.13.4 und 10.2	applikationsspezifisch, McuWIN.EXE	
49	zusätzliche Fehlerinfo bei Warnungen (Bitcode)	T1 McuWIN.EXE	
50ff	Oelmaier CNC-7A: Eingangsvariable für AWS		
50	Bei MBAWIN: ToolButton Manuell schalten	MbaWIN.EXE	
58	frei – nicht mehr belegt ab 29.01.2015		
59	AliveCounter McuWIN.exe -> RWMOS / SAP	McuWIN.EXE, Task1.src	

Nr.	Bedeutung	Programm	
60	Synchronisationsvariable für Programmstart 1 = Task 1 muss warten -1 = Task 1 ist in Endlosüberwachung	McuWIN.EXE, TASK1.SRC	
61	Referenzschalter ist Latcheingang (bitcodiert)	McuWIN.EXE, TASK0.SRC	
62	Keine Referenzierung auf Endschalter (bitcodiert)	McuWIN.EXE, TASK0.SRC	
63	Register zur Programmablaufsteuerung (bitcodiert)	McuWIN.EXE	
64	Bitweise Verschlüsselung der Spindeldaten	McuWIN.EXE, TASK0.SRC	
65	Spindelmaximaldrehzahl in User-Einheiten	McuWIN.EXE, TASK0.SRC	
Nr.	Bedeutung	Programm	
66	OverrideSteuerung für Gewindeschneiden Bit 0 = Verfahr-Override-Slider gesperrt Bit 1 = Spindel-Override-Slider steuert Spindel + Achsen	TASK0.SRC beschreibt McuWIN.EXE wertet aus	
67	BoardType (von InitMcuSystem3)	McuWIN.EXE	
69	Bitweise – Achsen, die im Open-Loop betrieben werden	McuWIN.EXE, TASK1.SRC	
70	Unbekanntes G-Kommando (bei Runtime Fehler 1)	TASK0	
71	Unbekanntes M-Kommando (bei Runtime Fehler 1)	TASK0	
72	Unbekanntes Kommando (bei Runtime Fehler 1)	TASK0	
73	Unbekanntes Kommando in CI3 (bei Runtime Fehler 1)	McuWIN.EXE	
90	Versionsfehler spezifizieren: 10 / 20 = Version von Task 0 ist zu klein für Task 1 11 / 21 = Version von Task 0 ist zu klein für Task 1 12 / 22 = Version von Task 0 ist zu klein für Task 1 14 / 24 = Version von RWMOS ist zu klein für Task 1 15 / 25 = Version von McuWIN ist zu klein für Task 1		
100 ff	DDE Zustandsvariable schreibend	McuWIN.EXE	
200 ff	DDE Zustandsvariable lesend	McuWIN.EXE	
300	Zustandsinfo manuelles Joystickverfahren bitcodiert Bit 0..7: Aktive Achsen bitcodiert Bit31: Verfahren freigeschaltet (TeachIn Maske ist sichtbar)	TASK1.SRC	
301	zeigt aktiven Achskanal an, ist notwendig für sicheren Achsstopp		
302	zeigt an, ob im SingleStep-Modus manuell verfahren werden kann (ab V2.5.3.122)	McuWIN.EXE, Task1.src	
310 .. 317	Nullpunkt-Offset Analogeingang für Analog Joystick Index-Einerstelle = Index der Achse und Index des Analogkanals	TASK1.SRC	
320 .. 327	Maximalauslenkung des Analogkanals, negatives Vorzeichen bewirkt Richtungsumkehr Index-Einerstelle = Index der Achse und Index des Analogkanals	TASK1.SRC	
390 .. 397	Debug-Anzeige AnalogIn kann verwendet werden zum Einrichten	TASK1.SRC	
500	Bei MBAWIN: Buttons Visible	MbaWIN.EXE	
501	Bei MBAWIN: Buttons Enable	MbaWIN.EXE	
600 bis 699	Applikationsspezifisch		

7 Beschreibung der Betriebsarten

7.1 EditMode

Betriebsart zum Editieren von Programmen. Programmstart ist möglich, wenn die Achsen referenziert sind (falls diese Eigenschaft nicht deaktiviert ist).
Im EditMode können die Achsen per TeachIn-Fenster manuell verfahren werden. Das manuelle Verfahren ist weiterhin über die CI-Variable CI39 möglich.

Tabelle: Bitcodierung in CI39:

Bit	Achse	Richtung
0	1	positiv
1	1	negativ
2	2	positiv
3	2	negativ
4	3	positiv
5	3	negativ
	usw.	

Das Beschreiben von CI39, abhängig von Digital-Eingängen kann z.B. in der Task2 in einer Endlosschleife erfolgen.

7.2 RunMode

Ein Programm wird im Automatikmodus ausgeführt. Referenzierung und erneuter Programmstart ist nun nicht möglich. Das Programm kann auch nicht editiert werden.
Mit Stop kann angehalten werden, mit Schritt kann in den StepMode übergegangen werden und mit Reset kann die Steuerung zurückgesetzt werden.
Der RunMode beendet sich selbst, wenn die UserTask und die Funktionstask stehen.
Mit den Kommandos M00 und M01 kann in den Step-Mode geschaltet werden. M01 ist nur wirksam, wenn der Button „Wahlweiser Halt“ aktiviert ist.

7.3 StepMode

Einzelschritt-Modus: Dieser beendet sich selbst, wenn nach einem Schritt keine neue Zeile mehr erreicht wird. Manuelles Verfahren der Achsen (per CI39) ist möglich, wenn der Merker „Manual Move in SS“ auf der Registerkarte „Desktop / System“ in IniCFG.exe gesetzt ist (Ini File Section [TEACHIN] – „ManualMoveInSS“ → CI302 <> 0).

7.4 HaltMode

Übergangszustand zwischen RunMode und EditMode.

7.5 ManMode

Manuelles Verfahren ist aktiv, (Teach-In, Referenzfahrt). In diesem Modus dürfen keine Automatikzyklen gestartet werden.

7.6 Referenzfahrt

Der Ablauf der Referenzfahrt kann mit den entsprechenden Variablen in McuWIN.INI konfiguriert werden. Falls sich die Referenzfahrt mit den vorgegebenen Variationsmöglichkeiten nicht konfigurieren lässt, kann im File „KdRefFahrt.INC“ ein Applikationsspezifischer Ablauf programmiert werden. Um diesen zu aktivieren muss in der Variable CI28 der Wert 1 eingetragen werden (z.B. im File „AppCommands.INC“).
Die Achsen werden immer sequentiell referenziert, in der Reihenfolge, die in der Variable „Reihenfolge“ angegeben ist.
Zunächst werden nun die eingestellten Werte für Jog-Geschwindigkeit, Jog-Beschleunigung, Home-Geschwindigkeit und Home-Beschleunigung überprüft. Falls hier ein Fehler erkannt wird (Wert <= 0) wird der Fehler „Konflikt in den Konfigurationsdaten!“ (20H) angezeigt.
Falls bei der Reihenfolge eine Achse mehrfach angegeben wird oder falls eine Achse angegeben wird, die nicht im System vorhanden ist, wird dieser Fehler ebenfalls angezeigt. Dann wird die Achse, falls erforderlich vom Endschalter freigefahren. Nun wird die Achse in die Suchrichtung gestartet, die in der Variable „Richtung“ in dem der Achse entsprechenden Bit angegeben ist. 1 bedeutet Suche in negative Richtung (zu kleineren Positionswerten).

7.6.1 Referenzierung auf Referenzschalter

Diese Variante wird achsspezifisch selektiert, wenn das der Achse entsprechende Bit in der Variable „Referenzschalter“ gesetzt ist. Dann wird die Achse in der angewählten Suchrichtung auf den Referenzschalter verfahren. Als Referenzschaltereingang **muss** der „schnelle Latcheingang“ der entsprechenden Achse verwendet werden (1. Achse = I14, 2. Achse = I15, 3. Achse = I16).

7.6.2 Referenzierung auf Endschalter

Diese Variante wird achsspezifisch selektiert, wenn das der Achse entsprechende Bit in der Variable „Referenzschalter“ nicht gesetzt ist. Dann wird die Achse auf den Hardware-Endschalter gefahren, und durch diesen gestoppt, und dann wieder freigefahren.

7.6.3 Feinpositionierung auf Nullspur

Durch Setzen des achsspezifischen Bit in der Variable „Indexsuche“ wird nun eine Feinpositionierung auf die Nullspur des Inkrementalgebers durchgeführt.

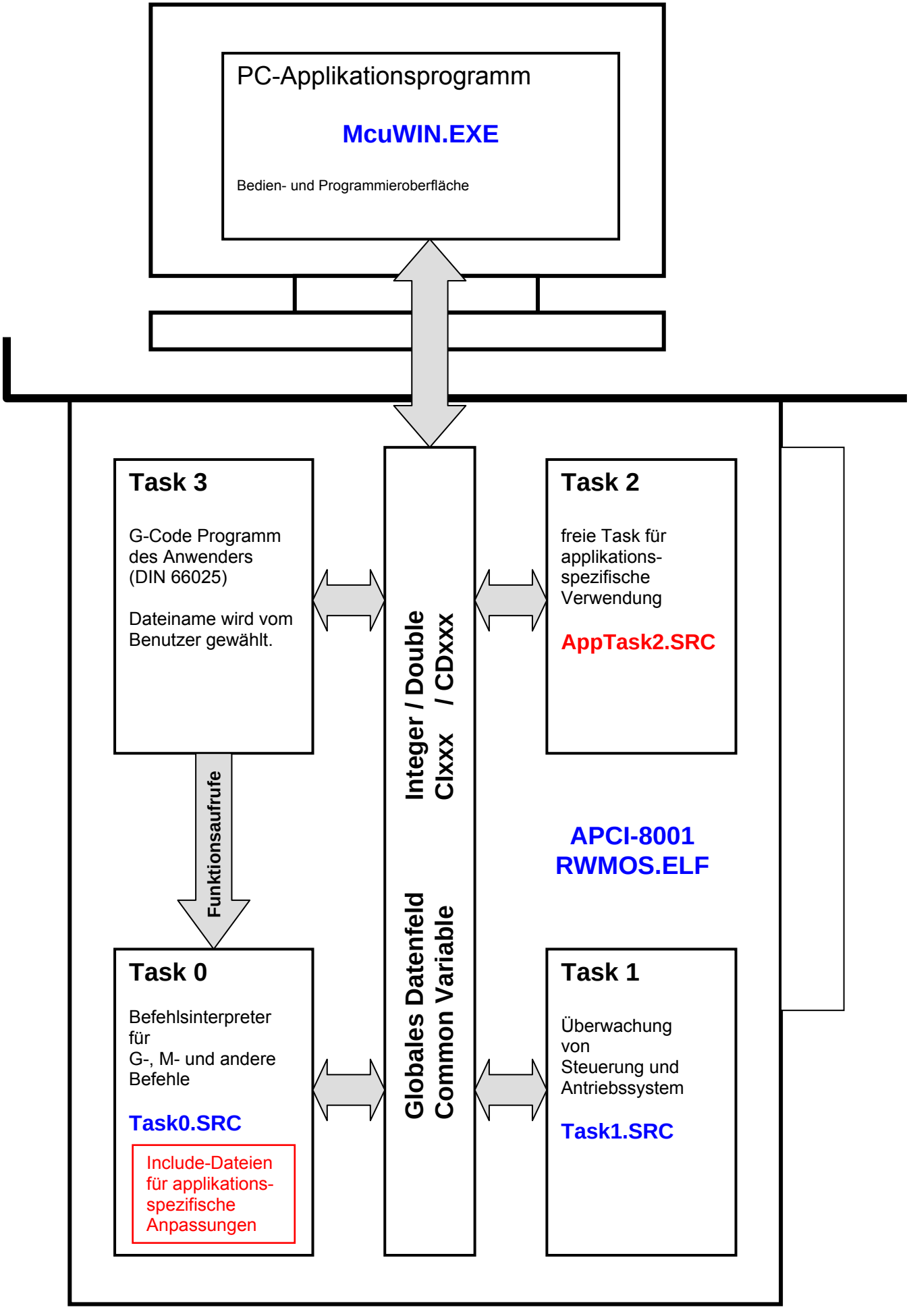
7.6.4 Homeposition setzen

In der nun gefundenen Position wird die Homeposition auf den Wert gesetzt, der in der Variable „ReferenzPos?“ für die entsprechende Achse angegeben ist. Nachdem alle Achsen referenziert wurden, wird die Hard- und Softwareendschalterüberwachung aktiviert.

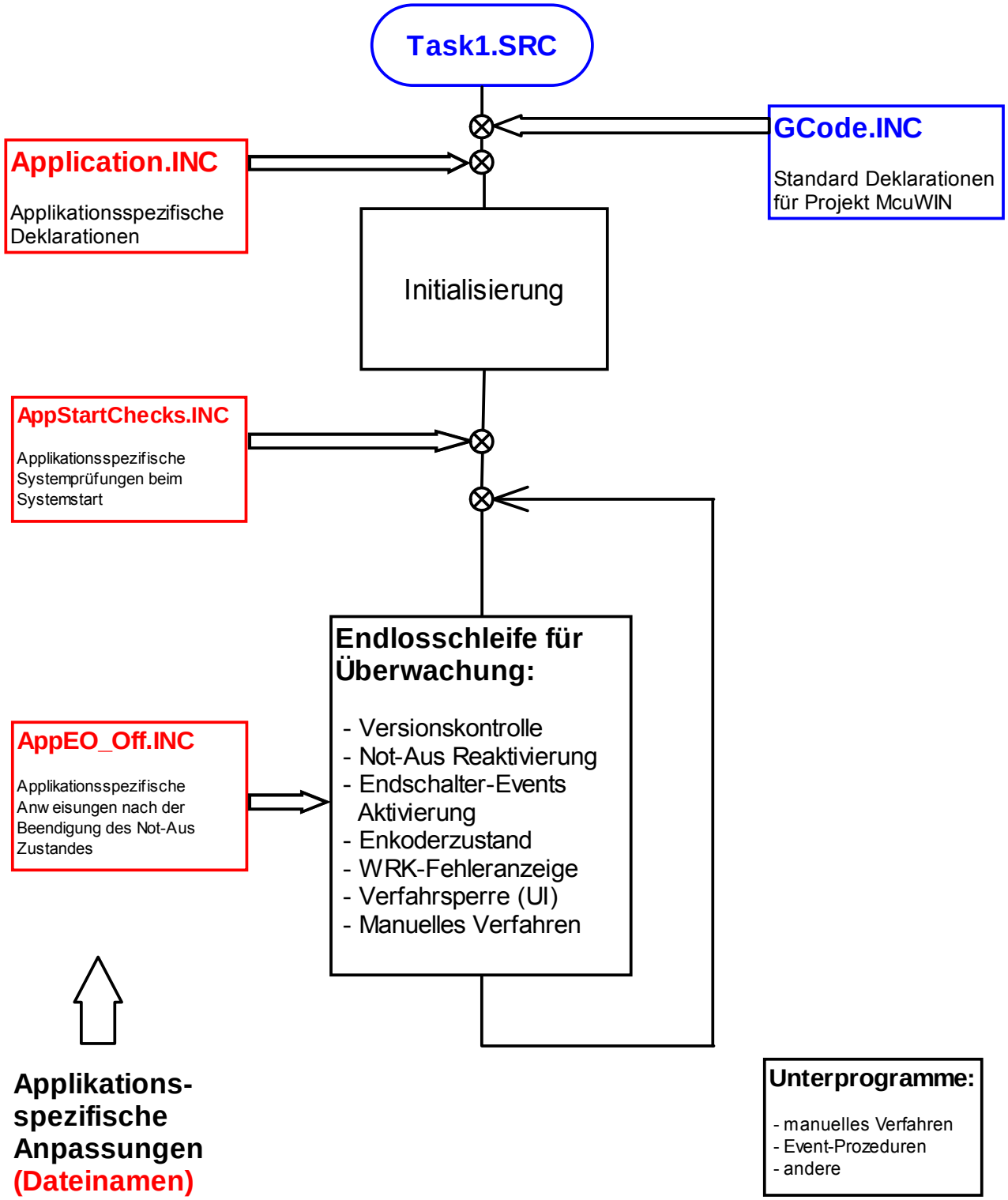
8 Beschreibung der SAP-Taskumgebung

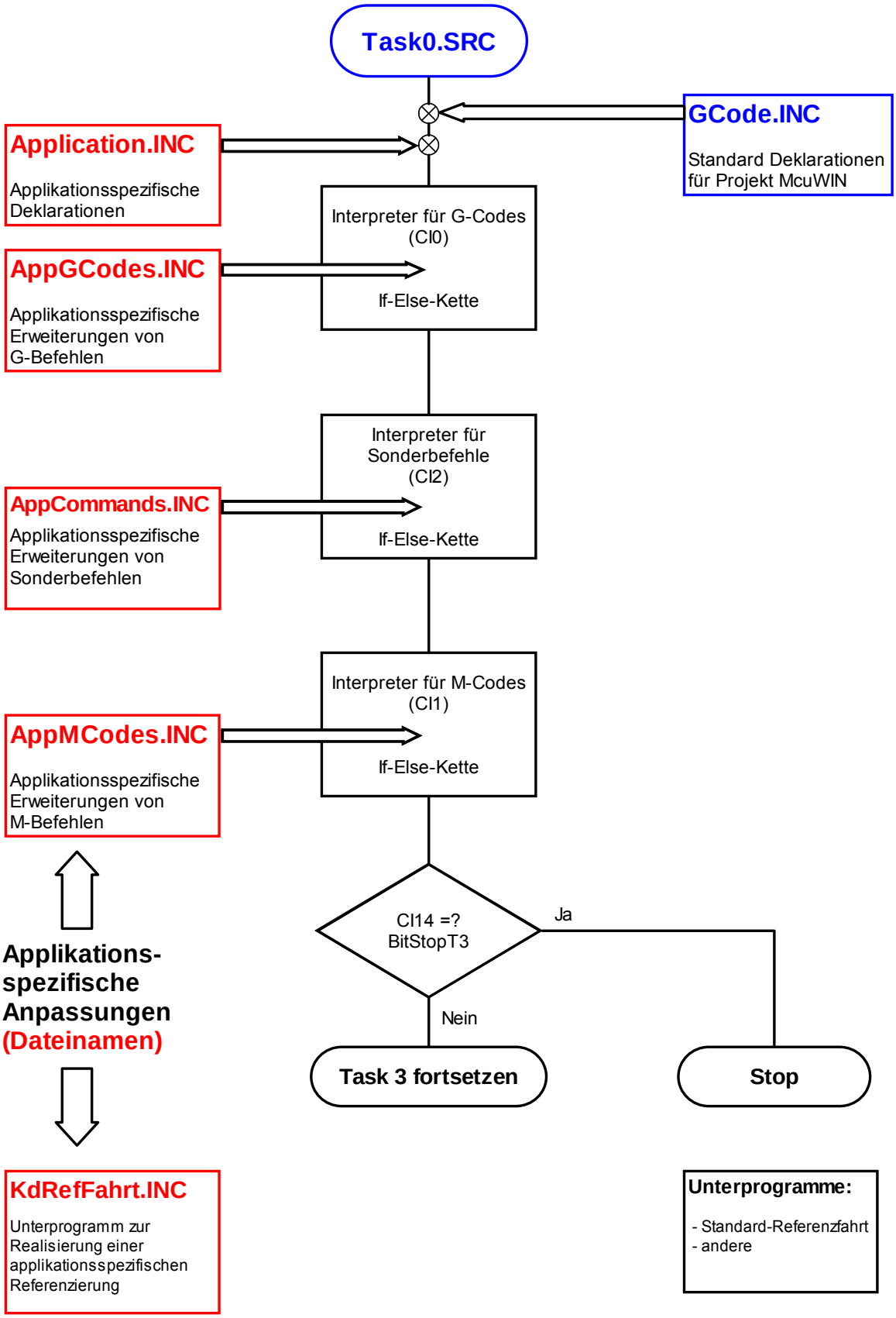
8.1 Hintergrundinformationen zur APCI-8001 / APCI-8008

G-Code-Programme nach DIN 66025 können mit der APCI-800x in einer SAP-Task abgearbeitet werden. Hierzu werden die Programme in eine CNC-Datei übersetzt, auf die Steuerung geladen und gestartet. Dabei ist der zur Verfügung stehende Befehlsumfang zu berücksichtigen, welcher applikationsspezifisch erweitert werden kann.
Wie im Handbuch beschrieben, stellt die APCI-800x vier Tasks zur Abarbeitung von NC-Programmen zur Verfügung. G-Code-Programme werden immer in der Task 3 ausgeführt. McuWIN verwendet diese Tasks wie in nachfolgendem Schaubild grafisch dargestellt.
Bei Änderungen in den Quelltext-Dateien bzw. in den Include-Dateien der SAP-Tasks müssen die Quelltext-Dateien jeweils neu übersetzt und danach auch auf der Steuerung durch Neustart von McuWIN wirksam gemacht werden. Das Übersetzen erfolgt mit dem Programm mcfg in einem Texteditor oder mit dem Kommandozeilenprogramm ncc.exe. Ein Übersetzen in McuWIN direkt ist nicht möglich, da dieses zur Verarbeitung von G-Code-Programmen eingestellt ist; die Hintergrund-Task-Programme sind jedoch in rw_SymPas programmiert.


PC-Applikationsprogramm
McuWIN.EXE
Bedien- und Programmieroberfläche
Task 3
G-Code Programm des Anwenders (DIN 66025)
Dateiname wird vom Benutzer gewählt.
Task 2
freie Task für applikations-spezifische Verwendung
AppTask2.SRC
Integer / Double
CIxxx / CDxxx
Globales Datenfeld
Common Variable
Funktionsaufrufe
APCI-8001
RWMOS.ELF
Task 0
Befehlsinterpreter für G-, M- und andere Befehle
Task0.SRC
Include-Dateien für applikations-spezifische Anpassungen
Task 1
Überwachung von Steuerung und Antriebssystem
Task1.SRC

In die Task 0 wird von McuWIN ein applikationsspezifischer Befehlsinterpreter geladen, der bei Bedarf von Task 3 aufgerufen wird. Weiterhin wird in Task 1 ein Initialisierungs- und Überwachungsprogramm gestartet, welches die Zuweisung von Standardwerten, wie z.B. Bahnbeschleunigung, Anfangwerte für CI- und CD-Variable und die Überwachung von Systemzuständen wie z.B. Endschalterbehandlung, Not-Aus-Behandlung usw. übernimmt. Ein Beispielprogramm zur G-Code-Programmierung steht unter dem Namen BEISPIEL.SRC zur Verfügung. Nachfolgend sind für Task 1 und Task 0 die Abläufe jeweils in einem Flussdiagramm dargestellt.




Task0.SRC
GCode.INC
Standard Deklarationen für Projekt McuWIN
Application.INC
Applikationsspezifische Deklarationen
Interpreter für G-Codes (CI0)
If-Else-Kette
AppGCodes.INC
Applikationsspezifische Erweiterungen von G-Befehlen
Interpreter für Sonderbefehle (CI2)
If-Else-Kette
AppCommands.INC
Applikationsspezifische Erweiterungen von Sonderbefehlen
Interpreter für M-Codes (CI1)
If-Else-Kette
AppMCodes.INC
Applikationsspezifische Erweiterungen von M-Befehlen
CI14 =? BitStopT3
Ja
Nein
Task 3 fortsetzen
Stop
Applikations-spezifische Anpassungen (Dateinamen)
KdRefFahrt.INC
Unterprogramm zur Realisierung einer applikationsspezifischen Referenzierung
Unterprogramme:
- Standard-Referenzfahrt
- andere

Die in den Diagrammen rot dargestellten Blöcke auf der linken Seite sind die applikationsspezifischen Erweiterungen. Diese werden bei Updates nicht ersetzt.

8.1.1 Compiler-Modus für G-Code-Programme

Das Programm McuWIN übersetzt die Programme im Editorfenster beim Start automatisch als Quelltextfiles im DIN66025 Modus (bei entsprechender Konfiguration von McuWIN. Diese Betriebsart kann auch bei den Compilern anderer Zusatzprogramme (z.B. für Offline-Übersetzung) angewählt werden.

8.1.1.1 Kommandozeilen Compiler NCC.EXE

Das Kommandozeilen-Programm NCC.EXE muss mit dem Parameter DIN aufgerufen werden.

Beispiel:

```
NCC PrgName DIN
```

8.1.1.2 MCFG für MS-DOS

Das Programm MCFG.EXE für DOS muss einmalig in den DIN-Mode umgestellt werden. Dies geschieht folgendermaßen:

Start des Programms mit dem Parameter PLC
Im Menü <<Editor>> <<Setup>> <<Set Compiler Mode>>
durch Mausklick die Option DIN-Code anwählen.
(Abschalten durch Anwahl von rw_SymPas)

Diese Information wird in der Datei MCFG.DAT abgespeichert!

8.1.1.3 MCFG für 32bit-Windows

Wenn ein Editorfenster geöffnet und aktiviert ist, kann im Menü <<Compile>> <<Options>> die Option <<Select DIN 66025 Language>> an oder abgewählt werden. Diese Information wird nicht gespeichert und muss nach jedem Neuaufruf aktiviert werden. Nach dieser Aktion steht im Menü <<Trace>> die zusätzliche Menüoption <<Trace DIN66025 Task>> zur Verfügung, mit der entsprechende Programme übersetzt, in die Task 3 geladen und gestartet werden.

8.1.2 Programmstatus Informationen der Steuerung

RWMOS.ELF stellt ab V2.5.3.50 diverse Status Informationen während des Ablaufs eines G-Code Programms zur Verfügung. Diese Informationen können zyklisch gelesen und zur Anzeige gebracht werden. Zum Lesen dieser Informationen ist mcug3.dll ab Version 2.5.3.34 und ein entsprechend aktuelles Hochspracheninterface erforderlich. Die entsprechenden Befehle sind nachfolgend beschrieben. Im Handbuch PHB ist diese Beschreibung nicht enthalten, da die Verwendung dieser Befehle ausschließlich im Zusammenhang mit dem G-Code Interface erfolgt. Die Kenntnis dieser Informationen ist nur dann wichtig, wenn eine eigene Programmieroberfläche oder Zustandsanzeige programmiert werden soll.

8.1.2.1 Informationen über aktuelles Verfahrprofil

Im „Profile Information Register“ {pir} werden bitcodiert folgende Informationen angezeigt:

Tabelle: Bitcodierung des pir

Bit / hex	Bezeichnung	Beschreibung
0 / 0001	***Linear***	Linear- oder Zirkularinterpolation wird ausgeführt.
1 / 0002	***Circular***	Zirkularinterpolation wird ausgeführt. Mit diesem Bit ist, i.A. auch Bit 0 gesetzt, da G02 und G03 Verfahrkommandos stets als Helixinterpolation ausgeführt werden.
2 / 0004	***Spline***	Spline Interpolation wird ausgeführt
3 / 0008	***Jog***	G00 wird ausgeführt. Dieses Bit wird auch bei Bewegungen während der Referenzierung gesetzt.
4 / 0010	***Absolut***	Gesetzt bei Absoluter Positionierung, wenn 0 dann Relativpositionierung
5 / 0020	***ClockWise***	G02 Kommando wird ausgeführt. Mit diesem Bit ist immer auch Bit Circular gesetzt. Wenn Bit Circular gesetzt ist und dieses Bit ist 0, dann wird ein G03 Kommando ausgeführt.
6 / 0040	***Spooled***	Dieses Bit zeigt an, daß gespoolte Verfahrkommandos ausgeführt werden (G01, G02, G03).
weitere	***undef.***	Diese Bits sind undokumentiert bzw. frei für zukünftige Erweiterungen.

Das Auslesen des „Profile Information Registers“ erfolgt mit der dll-Funktion rdpir:

BESCHREIBUNG:	Mit Hilfe dieses Befehls kann das „Profile Information Register“ (pir) gelesen werden. Die Bitcodierung des Registers ist aus obiger Tabelle ersichtlich.
BORLAND DELPHI:	function rdpir (var value: integer) : integer;
C:	int rdpir (int *value);
VISUAL BASIC:	Function rdpir (value As Long) As Long
Rückgabewert:	< 0: Kommando wird von RWMOS.ELF nicht unterstützt = 0: Kommando erfolgreich ausgeführt > 0: Zeitüberschreitung, Kommando nicht ausgeführt
ANMERKUNG:	Unter rw_SymPas kann auf pir mit der Systemvariablen *pir* zugegriffen werden.

8.1.2.2 Informationen über die achsspezifische Zielposition

BESCHREIBUNG:	Mit Hilfe dieses Befehls kann die achsspezifische Zielposition des aktuellen Profils (ptp) als Absolutwert in der achsspezifischen Einheit in die Variable value eingelesen werden. Mit dem Parameter *an* wird der Achsindex des auszulesenden Achskanals (0..n) angegeben. Hierbei wird jeweils die Zielposition des aktuell ausgeführten Teilprofils auch innerhalb einer Interpolationskontur ermittelt. Im Gegensatz dazu liefert die Funktion rdtp() nur die Zielposition der gesamten Kontur zurück.
BORLAND DELPHI:	function rdptp (an: integer; var value: double) : integer;
C:	int rdptp (integer an, double *value);
VISUAL BASIC:	Function rdptp (ByVal an As Long, value As Double) As Long
Rückgabewert:	< 0: Kommando wird von RWMOS.ELF nicht unterstützt = 0: Kommando erfolgreich ausgeführt > 0: Zeitüberschreitung, Kommando nicht ausgeführt
ANMERKUNG:	Unter rw_SymPas kann auf ptp mit dem Achsenqualifizierer ptp zugegriffen werden.

8.1.2.3 Informationen über die programmierte Verfahrgeschwindigkeit

BESCHREIBUNG:	Mit Hilfe dieses Befehls kann die programmierte und die tatsächlich realisierte Bahngeschwindigkeit des aktuellen Interpolations-Profils (dtv) als Absolutwert in der programmierten Interpolations-Einheit in das Array value eingelesen werden. Bei G00 wird jeweils der Wert 0 zurückgegeben
BORLAND DELPHI:	function rddtv (var value: array of double) : integer;
C:	int rddtv (double value[2]);
VISUAL BASIC:	Function rddtv (ByRef value As Double) As Long
Rückgabewert:	< 0: Kommando wird von RWMOS.ELF nicht unterstützt = 0: Kommando erfolgreich ausgeführt > 0: Zeitüberschreitung, Kommando nicht ausgeführt
ANMERKUNG:	Unter rw_SymPas kann auf dtv[] mit den Systemvariablen *dtv0 und dtv1* zugegriffen werden. **Vorsicht:** Der Funktion muss ein Array mit mindestens zwei Elementen übergeben werden, da ansonsten ein undefinierter Speicherbereich beschrieben wird. Im ersten Element wird die vom Anwender programmierte Bahngeschwindigkeit zurückgeliefert. Im zweiten Element wird die evtl. begrenzte Bahngeschwindigkeit zurückgeliefert.

8.2 Ergänzung von kundenspezifischen Codes

Die Liste der G- und M-Befehle kann vom Anwender applikationsspezifisch erweitert werden. Für die Auswertung der Parameter stehen die Variablen NUMPARAM, AXSEL und Common-Double zur Verfügung. Folgende Parametertypen sind möglich:

Liste mit Zahlenwerten: Die Zahlenwerte werden in CD0, CD1 etc. an die Taskumgebung übergeben. Die Anzahl dieser Parameter steht in der Variablen NUMPARAM. Die Variable AXSEL ist Null.

Beispiel: M24 20.6 7.3

AXSEL = 0
NUMPARAM = 2
CD0 = 20.6
CD1 = 7.3

Liste mit Achsbezeichnern: Hier gibt es keine Zahlenwerte. In der Variablen AXSEL werden die angegebenen Achsen bitweise codiert mit ihren Achsindices angezeigt. NUMPARAM ist die Anzahl der angegebenen Achsen. In den CD-Variablen mit dem Index der jeweiligen Achse wird der Index in der Reihenfolge der Angabe eingetragen (mit 0 beginnend). Dadurch kann die Reihenfolge einer Achsliste in einem G- oder M-Befehl ausgewertet werden.

Beispiel: M24 X Z

Die 1. Achse sei die X-Achse, die dritte Achse sei die Z-Achse
AXSEL = 5
NUMPARAM = 2
CD0 = 0.0
CD1 ist undefiniert
CD2 = 1.0

Liste mit Achsbezeichnern, denen ein Zahlenwert zugeordnet ist: In der Variablen AXSEL werden die angegebenen Achsen bitweise codiert mit ihren Achsindices angezeigt. NUMPARAM ist Null. Die Zahlenwerte werden in CD0, CD1 etc. an die Task-Umgebung übergeben, und zwar jeweils in der CD-Variablen, deren Index dem Achsindex entspricht. Mit Hilfe dieser Parameterübergabe können Achsen angewählt und mit einer Positions- oder Geschwindigkeitsangabe versehen werden.

Beispiel: M24 X20.6 Z7.3

Die 1. Achse sei die X-Achse, die dritte Achse sei die Z-Achse
AXSEL = 5
NUMPARAM = 0
CD0 = 20.6
CD2 = 7.3

8.2.1 G-Codes

Der Compiler verarbeitet nur einige G-Codes direkt. G-Codes, die nicht direkt vom Compiler verarbeitet werden, bewirken einen Aufruf der Task 0 mit Übergabe von Parametern in CI- und ggf. CD-Variablen. In der Task 0 kann die entsprechende Funktionalität des Befehls realisiert werden, sofern diese nicht bereits von der Steuerung bereitgestellt wird.
Nachfolgend aufgelistete G-Codes werden von der Betriebssystemsoftware RWMOS.ELF direkt verarbeitet und bewirken keinen Aufruf von Task 0:

G00, G01, G02, G03, G04, G53 – G60, G7, G71, G90 – G94, G150, G151.

Alle anderen G-Kommandos bewirken einen Aufruf der Task 0. Deren Funktionalität muss dort ausprogrammiert werden. Die Kommandos
G17-G19, G21 – G24, G39 – G42, G50, G51, G74, G98, G99, G161 und G162
sind in Task0.SRC programmiert und können, bei Bedarf, benutzerspezifisch reprogrammiert werden.
Einige G-Codes bewirken die Ausführung einer Bewegungskontur mit Hilfe von Spoolerbefehlen. Dies sind die Befehle
G01, G02, G03, G17, G18, G19, G40, G41, G42, G90, G91, G200 - G299.

Durch alle anderen G-Befehle wird eine Kontur beendet und ausgeführt. Bei Ausprogrammierung dieser G-Codes durch den Anwender, muss darauf geachtet werden, dass diese Kommandos auch mit Spoolerbefehlen belegt werden, insbesondere bei den noch freien Funktionen G200 bis G299.

Bei Ausführung eines unbekannten G-Codes wird Task 0 aufgerufen mit der G-Code Nummer in CI0. Eine Zuweisung an X, Y oder Z wird in CD0, CD1 oder CD2 übergeben (Index der CD-Variable = Index der angegebenen Achse, siehe oben). Die Achsen, die beim Befehlsaufruf angegeben wurden, werden in der Systemvariablen AXSEL bitcodiert angezeigt. Somit kann eine Überprüfung der Parameterliste erfolgen.

8.2.2 M-Codes

Auch einige M-Codes werden von RWMOS.ELF direkt verarbeitet. Dies sind die Kommandos

M17, M26, M27, M80, M96, M97, M98.

Alle anderen M-Kommandos bewirken einen Aufruf der Task 0. Deren Funktionalität muss dort ausprogrammiert werden. Die Kommandos

M00 - M06, M08, M09, M30,

sind in Task0.SRC programmiert und können, bei Bedarf, benutzerspezifisch reprogrammiert werden.
Die Ausführung einer Bewegungskontur wird mit den Befehlen

M00, M01, M26, M27 und M200 – M299 nicht unterbrochen. Bei Ausprogrammierung dieser M-Codes durch den Anwender, muss darauf geachtet werden, dass diese Kommandos auch tatsächlich mit Spoolerbefehlen belegt werden, insbesondere bei den noch freien Funktionen M200 bis M299.

Bei Ausführung eines unbekannten M-Codes wird Task 0 aufgerufen mit der M-Code Nummer in CI1. Dieses Kommando kann nun mit verschiedenen Parametertypen aufgerufen werden (siehe oben):

Aufruf mit Achszuweisungen, wie bei G-Codes:

Beispiel (Achse X = 1. Achse, Z = 3. Achse):
M37 X20 Z40

In diesem Fall zeigt die Systemvariable AXSEL bitcodiert die angegebenen Achsen an (AXSEL = 5). In CD0 ist der Wert 20, in CD2 der Wert 40 enthalten.

Eine Angabe von Achsenqualifizierern ohne Zahlenwert ist ebenfalls möglich:

M37 X Z
In diesem Fall wird in den entsprechenden CD-Variablen der Index in der Parameterangabe von 0 beginnend übergeben.

Aufruf mit Double-Parametern ohne Achszuweisung:

Beispiel:
M27 20.5

In diesem Fall zeigt die Systemvariable NUMPARAM die Anzahl der Parameter an (NUMPARAM = 1). In CD0 ist der Wert 20,5 enthalten. Etwaige weitere Parameter wären in den folgenden CD-Variablen enthalten. Der Datentyp des Parameters ist immer Double (Gleitpunktzahlen). Hierbei ist zu beachten, daß auch Berechnungsausdrücke als Parameter übergeben werden können. Dadurch ist es notwendig negative Zahlenwerte zu klammern.

Beispiel:
M27 10 –5
Dieser Aufruf wird als ein Parameter mit dem Wert (10-5) interpretiert. Wenn zwei Parameter übergeben werden sollen muss der Aufruf folgendermaßen aussehen:

M27 10 (-5)

Ein Mix von Parametertypen (Achsenqualifizierer und Zahlenwerte) ist nicht erlaubt.

8.2.3 Zustandsinformationen

Für die Implementierung eigener G- oder M-Codes ist oftmals die Kenntnis unterschiedlicher Systemzustände erforderlich. Nachfolgend wird der Zugriff auf einige wichtige Systemvariablen beschrieben:

8.2.3.1 Interpolationsebene

Die Werkzeugradius- und Werkzeuglängenkorrektur basiert stets auf einem rechtshändigen kartesischen Koordinatensystem. Hierbei wird vorausgesetzt, dass die Maschinenachse X dem Achsindex 0 (Erste Steuerungsachse) zugeordnet ist, die Maschinenachse Y dem Achsindex 1 (Zweite Steuerungsachse) und die Maschinenachse Z dem Achsindex 2 (Dritte Steuerungsachse). Falls hier eine andere Zuordnung verwendet werden soll, müssen in der Datei GCode.inc die Konstanten NrXAxis, NrYAxis und NrZAxis angepasst werden und danach muss die Quelltextdatei Task1.src übersetzt werden.

Die mit G17/18/19 gesetzte Interpolationsebene kann in der Objektvariablen ***FTcPlane_r*** gelesen werden. Diese enthält den Wert 17, 18 oder 19 und hat die Bedeutung laut entsprechendem G-Befehl.
Der Index der X-Achse kann der Objektvariablen ***FTcXAxNr_r*** entnommen werden.
Der Index der Y-Achse kann der Objektvariablen ***FTcYAxNr_r*** entnommen werden.
Der Index der Z-Achse kann der Objektvariablen ***FTcZAxNr_r*** entnommen werden.

8.2.3.2 Interpolationsachsen

Die Achsen, die aktuell als Interpolationsachsen definiert sind, sind in den niedrigstwertigen 16 Bit der Systemvariablen IPolMode abgebildet (siehe hierzu auch Kapitel 10.2.1).

8.2.4 Applikationsspezifische Verwendung der Default-Ausgänge

In der Datei „GCode.inc“ sind einige Ausgänge der Grundplatine für eine standardmäßige Verwendung vorgesehen. Diese Ausgänge sind auch in der Beschreibung von McuWIN aufgeführt. Wenn sie applikationsspezifisch verwendet werden sollen, müssen alle Funktionen, die diese Ausgänge verwenden, gepatcht werden. In nachfolgender Tabelle sind die Ausgänge, die Funktionen und die entsprechenden Quelltext-Dateien aufgelistet.

Tabelle: Default-Ausgänge und Hinweis für Patch

Signalname	Nr.	Verw. in Funktion	.. File	Patch in File
`DefaultKuehlung`	4	M08 / M09	Task0.SRC	AppMCodes.inc
`DefaultRichtungHS`	6	M03 / M04 SpindelStopp - M00 - M05 - M30 - 1000	Task0.SRC	AppMCodes.inc Application.inc AppMCodes.inc AppMCodes.inc AppMCodes.inc AppCommands.inc
`DefaultHauptspindel`	7	M03 / M04 SpindelStopp - M00 - M05 - M30	Task0.SRC	AppMCodes.inc Application.inc AppMCodes.inc AppMCodes.inc AppMCodes.inc
`DefaultProgrammAktiv`	8	G74 1000 1001 1002	Task0.SRC	AppGCodes.inc AppCommands.inc AppCommands.inc AppCommands.inc

8.2.5 Sonderfunktionen

Bei Ausführung eines unbekannten Sonderbefehls wird Task 0 aufgerufen mit der einer Befehls-Nummer in CI2. Ein Parameter wird in CD0 übergeben.

8.2.6 Weitere Aufrufkonventionen

Nach dem Aufruf von Task 0 z.B. durch einen selbstdefinierten G- oder M-Befehl, wird die Ausführung von Task 3 gestoppt. Task 0 muss nun den angezeigten Befehl abarbeiten. Nach Beendigung der Ausführung muss der Befehl quittiert und Task 3 fortgesetzt werden mit der Anweisung

contcnct (3);

Im SRC-Beispielprogramm ist weiterhin folgenden Konvention enthalten:

- Wenn nach der Befehlsausführung Task 3 wieder gestartet werden soll muss in CI14 das Bit 1 zurückgesetzt sein.
- Wenn nach der Befehlsausführung Task 3 nicht wieder gestartet werden soll wird in CI14 der Wert 1 eingetragen. Dieser Wert wird jew. wieder nach Befehlsausführung zurückgesetzt.

8.2.7 Fehler und Warnungen

Wenn im Programmablauf Fehler auftreten, kann ein Bit in der Fehlervariablen CI10 oder in CI48 gesetzt werden. Die Verwendung dieser Bits muss sorgfältig dokumentiert werden, um eine Doppelbelegung zu verhindern. Einige Bits in CI10 sind bereits vom System belegt. Diese sind in GCode.inc dokumentiert bzw. deklariert. Um einen Konflikt mit anwenderspezifisch belegten Bits nach zukünftigen Updates zu vermeiden, sollte vom Anwender nur auf die Fehlervariable CI48 zugegriffen werden. Wenn ein Bit in diesen Variablen gesetzt wird, wird eine Klartextmeldung im Fehlerfenster der Oberfläche angezeigt. Der Meldungsinhalt kann in McuWIN.INI in den Sektionen [FEHLERTEXTE] (für CI10) und [FEHLERTEXTE48] (für CI48) definiert werden. Beim Ändern bzw. bei der Neuanlage von Zeilen sind hier die entsprechenden Syntaxanforderungen einzuhalten. Am besten orientiert man sich an anderen Zeilen in der jeweiligen Sektion (siehe hierzu auch Kapitel 5.7). Wenn der entsprechende Fehler einen Programmstopp bewirken soll, sind entsprechende Operationen einzuleiten. Ein ordnungsgemäßer Programmstopp kann innerhalb Task 0 (und der entsprechenden Include-Dateien) z.B. mit folgender Sequenz erreicht werden:

```
CI0 := 0;
CI1 := 0;
CI2 := 1000;      // Stop-Kommando in Task 0 aufrufen
goto RestartT0;
```

Warnungen oder Zustandsmeldungen können durch Setzen eines Bits in CI15 generiert werden. Die Meldungstexte hierzu werden in der Sektion [WARNTEXTE] definiert. Ein aktives Bit in CI15 wird von der Oberfläche erkannt, angezeigt und gelöscht. Eine besondere Funktion hat das höchstwertige Bit in CI15: Wenn dieses Bit gesetzt wird, wird das Fehlerfenster gelöscht. Somit können vorher aufgelegte Anzeigen wieder gelöscht werden. In diesem Fall werden auch alle anstehenden Fehlermeldungen im Fehlerfenster gelöscht, die jedoch sofort wieder restauriert werden, wenn die entsprechenden Fehlerbits in den Fehlervariablen noch anstehen.

Hinweis: Um eine Warnungsanzeige sicher aufzulegen, darf keine unbearbeitete Clear-Anforderung in CI15 anstehen. Mit dem Unterprogramm

```
WaitForNoClearCI15;
```

wird abgewartet, bis dieses Bit den Wert 0 hat. Somit ist gewährleistet, dass die gewünschte Meldung auch tatsächlich angezeigt wird.

8.3 Task 0 – Befehlsinterpreter

In der Task 0 wird im Allgemeinen eine Task ausgeführt, welche die Funktionalität von G-Codes, M-Codes und von Sonderfunktionen bereitstellt, welche vom Compiler nicht direkt behandelt werden. Unter dem Namen TASK0.SRC liegt hierzu ein Programm vor, welches applikationsspezifisch ergänzt werden kann.

8.3.1 Implementierung kundenspezifischer G-Codes

Beim Aufruf eines G-Code-Kommandos, welches von der Steuerung nicht direkt verarbeitet wird, wird ein Funktionsaufruf in Task 0 durchgeführt. Dabei wird die Nummer des G-Codes in CI0 eingetragen. Werte die mit den Achsbezeichnern angegeben werden, werden in die Common-Double-Variable 0, 1, ... eingetragen, wobei der Index der CD-Variable dem Achsindex entspricht. Die Achsen, die beim Befehlsaufruf angegeben wurden, werden in der Systemvariablen AXSEL bitcodiert angezeigt. Danach wird die G-Code-Task (Task 3) angehalten und Task 0 gestartet. Die Ergänzung des Interpreters in Task 0 erlaubt somit das Ergänzen eigener G-Codes. Im Allgemeinen erhält der Anwender auf Wunsch eine entsprechend angepasste Version der Datei TASK0.SRC nach Angabe der Ergänzungswünsche. Der Anwender kann jedoch auch eigene Erweiterungen vornehmen. Hierzu ist das nachfolgende Kapitel zu beachten.

Hinweis: Nach einer Reinstallation von McuWIN muss TASK0.SRC neu übersetzt werden, wenn eines der nachfolgenden Include-Files kundenspezifische Ergänzungen enthält.

8.3.2 Include-Dateien in TASK0

8.3.2.1 GCode.INC

In dieser Datei sind projektweite installationsspezifische Konstanten und Variablendeklarationen enthalten. Diese Datei wird bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN ersetzt. Somit gehen etwaige benutzerspezifische Ergänzungen an dieser Stelle bei Updates verloren. Diese Datei sollte deshalb vom Anwender nicht modifiziert werden. Hierzu steht die Datei Application.INC zur Verfügung.

8.3.2.2 Application.INC

In dieser Datei können projektweite kundenspezifische Konstanten und Variablendeklarationen eingetragen werden.
Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar.

8.3.2.3 KdRefFahrt.INC

Der Ablauf der Referenzfahrt ist in TASK0.SRC in der Prozedur „Referenzfahrt“ programmiert. Wenn dieser Ablauf kundenspezifisch angepasst werden soll, sollte dieser Ablauf in der Include-Datei „KdRefFahrt.INC“ programmiert werden. Diese Funktion muss dann aktiviert werden durch Eintrag eines Wertes in CI28 <> 0. Diese Option kann im Einrichtprogramm IniCFG.EXE gesetzt werden. Vor McuWIN V2.5.3.24 musste dieser Eintrag in AppTask2.SRC gemacht werden.
Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar.

8.3.2.4 AppGCodes.INC

In dieser Datei werden applikationsspezifische G-Codes ausprogrammiert.
Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar.

8.3.2.5 AppCommands.INC

In dieser Datei werden applikationsspezifische Sonderbefehle wie z.B. S-Befehl ausprogrammiert. Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar.

8.3.2.6 AppMCodes.INC

In dieser Datei werden applikationsspezifische M-Codes ausprogrammiert. Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar.

8.4 Task 1 - Initialisierungs- und Überwachungstask

Die Task 1 steht zur Ausführung einer Initialisierungs- und Überwachungstask zur Verfügung. Ein Beispielprogramm hierzu steht unter dem Namen TASK1.SRC zur Verfügung. Diese Task kann applikationsspezifisch ergänzt werden. Diese Task arbeitet zunächst eine Initialisierungsliste ab und verbleibt dann in einer Endlosschleife, in der zyklische Überwachungsfunktionen abgearbeitet werden.
I.A. erhält der Anwender auf Wunsch eine entsprechend angepasste Version der Datei TASK1.SRC nach Angabe der Ergänzungswünsche.

8.4.1 Initialisierungen im Modul TASK1.SRC

Die nachfolgenden Ausführungen sind nur für die Systemanpassung wichtig. Der Programmbediener wird mit diesen Systemfiles nicht konfrontiert!

8.4.1.1 Einheit für Interpolationsbefehle

Hierzu wird den Systemvariablen TU bzw. PU eine Zeit- bzw. Positionseinheit zugewiesen (siehe Handbuch PHB Abschnitt 6.2.1.2)

8.4.1.2 Beschleunigung für Interpolationsbefehle

Um den Beschleunigungswert für Interpolationsbefehle zu setzen wird der Systemvariable TRAC ein entsprechender Wert zugewiesen (Handbuch PHB Abschnitt 6.3.1). Die Einheit dieses Wertes ist in den Systemvariablen PU und TU definiert.

8.4.1.3 CI- und CD-Variable

Dies sind applikationsspezifische Variable, die Systemgrößen, Zustands- und Fehlervariable enthalten können. Diese Variable können von einer PC-Oberfläche direkt gelesen und ausgewertet werden. Folgende Variable werden vom System verwendet z.B. zur Parameterübergabe bei Code-Aufrufen:

Tabelle: Common Variable, die für Parameterübergabe verwendet werden

Variable	**Verwendung**
CI0	G-Code Nr.
CI1	M-Code Nr.
CI2	Sonderbefehl Nr.
CI10	Fehlervariable
CI11	Referenzierte Achsen (bitcodiert)
CD0	Parameter für diverse Befehle
CD1	Parameter für diverse Befehle
CD2	Parameter für diverse Befehle
CD3..7	Parameter für diverse Befehle

8.4.2 Überwachungen

Die Überwachungsfunktionen werden in der Task 1 ausgeführt werden, welche ständig aktiv ist. Einzelne Überwachungen werden ggf. durch Systemzustände aktiviert und deaktiviert werden (siehe z.B. Software-Endschalter).
Einige Systemzustände werden von der RWMOS-Betriebssystemsoftware überwacht und lösen beim Auftreten eine entsprechende Event-Routine aus. Diese Routinen müssen vor Verwendung aktiviert werden (z.B. EVMPE). Hierzu sind die Beschreibungen im Handbuch PHB Abschnitt 6.4 heranzuziehen.
Vorsicht: In der Task, in der die Events behandelt werden, dürfen sich keine langen Wait-Kommandos befinden, da während eines Wait z.B. wt (2000); die entsprechende Task inaktiv ist.

8.4.2.1 Schleppfehler

Der achsspezifische maximale Schleppfehler wird im Programm mcfg definiert. Ein Überschreiten dieses Schleppfehlers kann in der Eventprozedur EVMPE behandelt werden.

8.4.2.2 Hardware-Endschalter

Ein Hardware-Endschalterereignis kann in der Eventprozedur EVLSH behandelt werden.

8.4.2.3 Software-Endschalter

Ein Software-Endschalterereignis kann in der Eventprozedur EVLSS behandelt werden. Dieses Ereignis wird erst nach dem Kommando SHP bei der jeweiligen Achse überwacht, da bei nicht referenzierten Achsen die Überwachung eines Software-Endschalters i.A. nicht sinnvoll ist.

8.4.2.4 Not-Aus

Wenn ein Eingang aktiv wird, dem die Funktion EO zugewiesen wurde, wird in die Eventbearbeitungsroutine EVEO verzweigt.
Diese Routine darf ein Not-Aus-Ereignis nur indirekt behandeln. Zur direkten Wirkung des Not-Aus-Schalters auf die Antriebe sind die entsprechenden Sicherheitsbestimmungen zu beachten.

8.4.2.5 Verstärker bereit

Die Event-Routine EVDNR wird ausgeführt, wenn ein Eingang inaktiv wird, welcher als DR deklariert wurde.

8.4.2.6 Enkoder-Error-Flag

Dieses Flag kann zur Erhöhung der Systemsicherheit zyklisch überwacht werden.

8.4.2.7 Enkoder-Verifikation

Mit Hilfe der Index-Latch-Funktion kann der Zählerstand der Enkoderlogik zyklisch überwacht werden.

8.4.2.8 Weitere Überwachungsfunktionen

An dieser Stelle können noch weitere Überwachungsfunktionen eingefügt werden. Hierzu ist die Dokumentation der APCI-8001 heranzuziehen.

8.4.3 Include-Files in TASK1

8.4.3.1 GCode.INC

Siehe hierzu Abschnitt 8.3.2.1

8.4.3.2 Application.INC

Siehe hierzu Abschnitt 8.3.2.2

8.4.3.3 AppStartChecks.INC

Dieses Modul wird in der Initialisierungsphase von Task1.SRC eingebunden und bietet die Möglichkeit für einmalige Initialisierungen und Systemüberprüfungen beim Systemstart oder nach einem System-Reset. Dies können z.B. Überprüfungen von I/O-Anforderungen wie Luft- oder Öldruck sein.
Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar.

8.4.3.4 AppEO_Off.INC

In dieser Datei können applikationsspezifische Anweisungen, z.B. das Setzen eines Freigabeausgangs, eingefügt werden, die nach einer Aufhebung des Not-Aus-Zustands ausgeführt werden sollen.
Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar.

9 Spindelsteigungs- und Winkelfehler-Kompensation

Spindelsteigungsfehler und Winkelfehler bei kartesischer Achsanordnung können in McuWIN auf einfache Weise kompensiert werden. Hierzu muss das McuWIN-Verzeichnis um einfache Textdateien, welche die Kompensationstabellen enthalten, ergänzt werden. Die Namen dieser Dateien sind von McuWIN vorgegeben. McuWIN prüft beim Start das McuWIN-Verzeichnis auf das Vorhandensein dieser Dateien und initialisiert gegebenenfalls die entsprechende Kompensation. Voraussetzung für die Fähigkeit des Systems, Kompensationstabellen zu verarbeiten, ist die Option optionELCAM in RWMOS.ELF. Die Namen dieser Dateien sind folgendermaßen aufgebaut:

CompTable_X_Y.txt

Y ist hierbei der symbolische Achsname der zu korrigierenden Achse; X ist der symbolische Name der Achse, über deren Verfahrbereich die Kompensation ermittelt wird. X und Y können auch gleich sein; in diesem Fall handelt es sich um die Korrektur eines Spindelsteigungsfehlers. Der symbolische Achsname ist der Name, der einer Achse in mcfg.exe bei den Systemdaten zugeordnet wird.

Eine Kompensationstabelle enthält eine Kopfzeile und die eigentliche Tabelle mit den zu kompensierenden Achsfehlern. Diese Tabelle enthält zeilenweise jeweils einen Achspositionswert und den Achsfehler an der Stelle des Positionswerts. Die Einheit dieser Werte ist die eingestellte achsspezifische Positionseinheit, z.B. mm.

Beispiel für eine Zeile einer Kompensationstabelle:

100.0 -0.015

Eine Kompensationstabelle muss aus mindestens zwei Stützpunkten (Zeilen) bestehen, kann aber auch mehrere Hundert Stützpunkte haben. Hierbei muss der Achspositionswert immer steigend angegeben werden. Die Kompensationswerte zwischen diesen Stützpunkten werden linear interpoliert. Der minimale und der maximale Achspositionswert sollen jeweils außerhalb des Verfahrbereiches der jeweiligen Achse liegen.

Wenn eine oder mehrere solcher Tabellen gefunden werden, erzeugt McuWIN beim Start die Protokolldatei „SpindleComp_Protocol.txt“, in welche Anmerkungen zur Programmierung der Tabelle geschrieben werden. Wenn ein Fehler bei der Konfiguration der Fehlerkompensation erkannt wird, wird dies durch ein Fehlerfenster angezeigt. In diesem Fall muss vor der Verwendung von McuWIN zunächst die entsprechende Datei korrigiert werden.

9.1.1 Spindelsteigungsfehler-Kompensation

Um einen Spindelsteigungsfehler zu kompensieren, muss eine Tabelle mit folgendem Aufbau vorliegen:

Dateiname:	CompTable_X_X.txt
Inhalt erste Zeile:	AnzahlStützpunkte
Inhalt folgende Zeilen:	Positionswert Kompensationswert

Beispiel: Der Spindelsteigungsfehler der Y-Achse soll kompensiert werden
Verfahrbereich der Achse von 0 .. 500 mm
Fehler bei 0: 0 mm
Fehler bei 200 mm +0,1 mm
Fehler bei 500 mm –0,05 mm

Dateiname: CompTable_Y_Y.txt

Datei-Inhalt:
5
-1000.0 0.0
0.0 0.0
200.0 0.1
500.0 -0.05
1000.0 -0.05

9.1.2 Winkelfehler-Kompensation

Bei der Winkelfehlerkompensation sind grundsätzlich zwei unterschiedliche Fälle möglich:
1.) Der zu kompensierende Fehler einer Achse ist von einer zweiten Achse (Führungsachse) abhängig.
2.) Der zu kompensierende Fehler einer Achse ist von der Position zweier weiterer Achsen abhängig.
Es gibt also eine erste und eine zweite Führungsachse.

Im ersten Fall erfolgt die Kompensation analog zur Kompensation eines Spindelsteigungsfehlers. Die Kompensationsdatei ist analog dazu aufgebaut, allerdings ist die Führungsachse mit der zu kompensierenden Achse nicht identisch.

Im zweiten Fall ist die Kompensationsdatei etwas umfangreicher. Nur dieser Fall wird nachfolgend noch behandelt. Hierzu wird eine sogenannte Multi-Line-Tabelle definiert. Es existiert praktisch eine ganze Reihe von Kompensationstabellen, die linear über den Verfahrbereich der zweiten Führungsachse verteilt werden. Zum besseren Verständnis dieses Zusammenhangs ein kleines Beispiel:

Kartesisches Koordinatensystem mit X, Y und Z-Achse
Die Position der Z-Achse soll kompensiert werden. Abhängig von X- und Y-Position ergeben sich unterschiedliche Werte der Kompensation. Deshalb müssen nun mehrere Tabellen, die den Kompensationswert von Z über der X-Achse beschreiben, definiert werden. Diese Tabellen werden auf den Y-Verfahrbereich verteilt. Der Bereich, über den diese Tabellen linear verteilt werden, wird mit den Positionswerten MLStart und MLEnd definiert (siehe nachfolgende Beschreibung des Datei-Inhalts).

Um einen Spindelsteigungsfehler mit einer Multi-Line-Tabelle zu kompensieren, muss eine Textdatei mit folgendem Aufbau vorliegen:

Dateiname: CompTable_X_Z.txt
X ist der symbolische Name der ersten Führungsachse, Z ist der symbolische Name der zu kompensierenden Achse

Inhalt erste Zeile: AnzahlStützpunkte AnzahlTabellen MLStart MLEnd MultiLineAchse
AnzahlStützpunkte ist die Anzahl der Werte in einer Tabelle (wie oben)
AnzahlTabellen ist die Anzahl der Multi-Line-Tabellen. Der erste und der letzte Positionswert einer Tabelle müssen immer gleich sein. Die Anzahl der Zeilen einer Tabelle muss ebenfalls immer gleich sein (AnzahlStützpunkte).
MLStart ist der Anfangswert der zweiten Führungsachse, MLEnd ist der Endwert der zweiten Fühungsachse; zwischen diesen beiden Positionswerten werden die einzelnen Kompensationstabellen gelegt.
MultiLineAchse ist die Achsnummer der zweiten Führungsachse; hier wird kein symbolischer Achsname verwendet, sondern die Nummer des Achskanals mit von 0 beginnender Zählweise.

Inhalt folgende Zeilen: Positionswert Kompensationswert
Aufbau wie bei der Spindelsteigungsfehler-Kompensation, allerdings werden hintereinander mehrere Tabellen eingetragen. Es folgen also (AnzahlStützpunkte x AnzahlTabellen) Zeilen in der Datei.

Beispiel: Der Fehler der Z-Achse soll kompensiert werden
Verfahrbereich der X-Achse von 0 .. 500mm
Verfahrbereich der Y-Achse von 0..100mm
Die Y-Achse ist die zweite Achse im System (Index 1)

Dateiname: CompTable_X_Z.txt

Datei-Inhalt:

```
5          3          0.0       100.0     1
-1000.0    0.0
0.0        0.0
200.0      0.1
500.0      -0.05
1000.0     -0.05
-1000.0    0.1
0.0        0.1
200.0      0.2
500.0      -0.05
1000.0     -0.05
-1000.0    0.0
0.0        0.0
200.0      0.2
500.0      -0.05
1000.0     -0.05
```

In diesem Beispiel hat jede Tabelle 5 Stützpunkte; jede Tabelle beginnt bei –1000 und endet bei +1000. Die Kompensationswerte an den Stützpunkten der einzelnen Tabellen unterscheiden sich teilweise. Die erste Tabelle ist gültig für einen Y-Wert von 0.0, die dritte (letzte) Tabelle ist gültig bei einem Y-Wert von 100.0. Die zweite (mittlere) Tabelle wird automatisch für den mittleren Y-Wert von 50.0 herangezogen. Zwischenwerte bei anderen Y-Werten werden automatisch linear interpoliert berechnet.

10 Kundenspezifische Erweiterungen und Updates

Die Quelltexte der CNC-Task Files werden mit dem Programmpaket ausgeliefert und stehen somit dem Anwender für eigene Ergänzungen und Anpassungen zur Verfügung. Um jedoch für Standardanwendungen eine Kompatibilität gegenüber zukünftigen Updates zu gewährleisten, können benutzerspezifische Anpassungen auf bestimmte Files beschränkt werden. Somit ist gewährleistet, dass Updates der wichtigsten Files (TASK0.SRC und TASK1.SRC) jederzeit aktiviert werden können. Um ein Update zu installieren, wird der aktualisierte Programmstand (Setup.exe) über die bestehende Installation in das gleiche Verzeichnis wie zuvor (ohne vorherige Deinstallation) installiert. Hierbei werden die nachfolgend beschriebenen applikationsspezifischen Files nicht aktualisiert. Nach jedem Update von McuWIN müssen die Quelltext-Files aller Tasks z.B. mit mcfg.exe in der jeweils aktuellen Version fehlerfrei übersetzt werden, damit die applikationsspezifischen Include-Files wirksam sind.
Zur Erinnerung: In Task 0 wird das Programm TASK0.SRC ausgeführt. Dieses enthält einen Befehlsinterpreter, in welchem G-Codes, M-Codes und weitere Funktionen programmiert sind.
In Task 1 wird das Überwachungs-Programm TASK1.SRC ausgeführt.
Der Anwender hat die Möglichkeit, kundenspezifische Erweiterungen / Anpassungen in den nachfolgend beschriebenen Files zu realisieren.
Vorsicht: Wenn andere als die unten angegebenen Dateien vom Anwender verändert wurden, müssen die entsprechenden Änderungen nach einem Update wieder manuell eingepflegt werden. Eine vollständige Dokumentation etwaiger Änderungen ist deshalb zwingend notwendig.

10.1 AppTask2.SRC

Dieses File wird in Task2 geladen und ausgeführt. Hier können kundenspezifisch einmalige Initialisierungen und in einer Endlosschleife zyklische Überwachungen implementiert werden.
Diese Datei bleibt bei einer Reinstallation (Neuinstallation ohne vorherige Deinstallation) von McuWIN erhalten. Somit bleiben benutzerspezifische Ergänzungen an dieser Stelle auch nach Updates verfügbar. Falls die Task2 beim Start von McuWIN automatisch gestartet werden soll, muss dies in IniCFG.EXE auf der Registerkarte „Dateien“ entsprechend angewählt werden.

10.2 Systemvariable und Sonderfunktionen

Nachfolgend werden Systemvariable und Funktionen beschrieben, die bei der Ausführung von G-Code-Programmen bzw. in der McuWIN-Umgebung hilfreich verwendet werden können, jedoch für die normale PCAP-Programmierung ohne Bedeutung sind.

Besondere Beachtung muss den Common-Variablen (CI0 bis CI999 und CD0 bis CD999) der Achsensteuerungskarten geschenkt werden. In der Standardversion werden diese Variablen verwendet, um einen Datenaustausch zwischen McuWIN.EXE und der Task-Umgebung bzw. zwischen den einzelnen Tasks auszuführen. Die Funktionsweise dieser Variablen darf keinesfalls durch Verwendung in eigenen Funktionen, Funktionserweiterungen oder Zugriffe aus der Anwenderebene heraus gestört werden, da ansonsten die Funktion des gesamten Pakets empfindlich gestört werden kann. Deshalb sind bei der Verwendung von Common-Variablen in eigenen Erweiterungen allerhöchste Disziplin und Sorgfalt notwendig. Insbesondere der Bereich von 0..99 ist in allen Fällen nahezu vollständig belegt. Auch derzeit scheinbar freie Variable können in zukünftigen Versionen (z.B. nach Updates) plötzlich in Verwendung sein und dürfen deshalb nicht von applikationsspezifischen Erweiterungen verwendet werden. Hierzu sind deshalb folgende Bereichsvereinbarungen einzuhalten:

Bereich	Common-Integer CI	Common-Double CD	Verwendung
0 .. 99	System	System	McuWIN Paket
100..199	DDE-Senden	-	nur im Zusammenhang mit DDE in applikationsspezifischen Erweiterungen
200.299	DDE-Empfang	-	nur im Zusammenhang mit DDE in applikationsspezifischen Erweiterungen
300.399	+	+	Analog-Joystick, reserviert für System
400..599	+	+	kann für applikationsspezifische Erweiterungen verwendet werden
600..999	+	+	frei für Anwender / Maschinenbediener

Ausnahmen hiervon sind CI15 und CI48, welche für applikationsspezifische Fehlerbehandlungen (CI48) bzw. für applikationsspezifische Warnanzeigen (CI15) verwendet werden können (siehe Kapitel 3.13.4) und CI58.

Um die Common Integer Bereiche (CI) unter 600 gegen Zugriffe von der Anwenderebene zu schützen, kann in IniCfg.exe auf der Registerkarte „Desktop / System“ die Einstellung „Common-Integer Variable < 600 nur lesen:“ aktiviert werden. Dieser Schutz ist standardmäßig nach einer Neuinstallation aktiv. Für die Common-Double Variable existiert dieser Schutz nicht.

10.2.1 Systemvariable IPOLMODE

Dies ist eine bitcodierte Variable, in welcher die Interpolationsachsen während der Laufzeit eines Programms hinterlegt sind. Diese sind in den niederwertigsten Bits der Variable abgelegt. Die höherwertigen 16 Bits der Variable enthalten Laufzeitinformationen und dürfen auf keinen Fall verändert werden.

Programmierbeispiel:

Die ersten 3 Achsen sollen als Interpolationsachsen in IPOLMODE hinterlegt werden.

```
IPOLMODE := (IPOLMODE and $FFFF0000) or $7;
```

11 Beispiele für applikationsspezifische Ergänzungen

11.1 Kühlmittel Ein/Aus (M08 / M09)

Mit dem Kommando M08 soll das Kühlmittel eingeschaltet werden. Dies geschieht mit Hilfe des Ausgangs 5 des ersten Achskanals. Hierzu wird AppMCodes.INC folgendermaßen ergänzt:

```
// Applikationsspezifische M-Befehle
if (CI1 = 8) then begin      // M08 - Kühlmittel Ein
      X.digob.5 := TRUE;     // Ausgang 5 von X-Achse einschalten
end else if (CI1 = 9) then begin   // M09 - Kühlmittel Aus
      X.digob.5 := FALSE;    // Ausgang 5 von X-Achse ausschalten
end else
```

Beachtenswert ist hier der Beginn der If-Else-Kette, der mit „end else“ abgeschlossen werden muss, damit die Standard-Befehlsinterpretation in TASK0.SRC, die ja mit if ()... beginnt, direkt an obige If-Else-Kette anschließt. Durch diese Vorgehensweise ist auch gewährleistet, daß Befehle die vom Anwender redeklariert werden, in der Anwenderprogrammierung zuerst erkannt und dadurch auch an dieser Stelle ausgeführt werden.

11.2 Hauptspindel

Mit dem S-Kommando soll die Spindeldrehzahl gesetzt werden. Hierbei sollen ein Minimal- und ein Maximalwert überwacht werden. Die Ausgabe der Spindeldrehzahl erfolgt über den Analogausgang des 4. Achskanals (H). Mit dem Kommando M03 soll die Spindel mit Rechtslauf eingeschaltet werden. Dies erfolgt durch den Ausgang 6 = EIN (Spindel Ein) und Ausgang 7 = 0 (Rechtslauf). Mit M04 soll die Spindel mit Linkslauf eingeschaltet werden. Dies erfolgt durch den Ausgang 6 = EIN (Spindel Ein) und Ausgang 7 = 1 (Linkslauf).
Mit M05 soll die Spindel ausgeschaltet werden.
Ein Erreichen der Spindeldrehzahl wird hier nicht überwacht. Ein Umschalten von Rechts- auf Linkslauf oder umgekehrt ist nicht möglich. Bei einem derartigen Aufruf wird ein der Fehler 00010000hex ausgegeben.

Hierzu wird AppMCodes.INC folgendermaßen ergänzt:

Um mit Hilfe des S-Kommandos einen Analogwert auszugeben wird AppCommands.INC folgendermaßen ergänzt:

```
if (CI2 = 1) then begin      // S-Kommando: Analogwert an Hauptspindel
//(Achse H) ausgeben
      // Zunächst Bereich des Analogwertes überprüfen
      // Minimalwert steht in CI601
      // Maximalwert steht in CI602
      if CD0 < CI601 then begin
            CI10 := CI10 or $00020000;    // Fehler in CI10 anzeigen
      end;
      if CD0 > CI602 then begin
            CI10 := CI10 or $00020000;    // Fehler in CI10 anzeigen
      end;
      if CI10 = 0 then begin // Wenn kein Fehler ansteht
            H.mcp := integer (CD0);       // Spindeldrehzahl auf
                                          // Analogausgang ausgeben
      end;
end else
```

12 DDE-Kommunikation

Das Programm McuWIN hat die Möglichkeit als DDE-Client Daten mit einer DDE-Serverapplikation auszutauschen. Um diese Funktionalität zu aktivieren müssen folgende Werte in der Datei McuWIN.INI definiert werden:
Section [DDE]
DDE Server- und Topic-Name:
ServerName=
TopicName=

Items zum Senden von Daten (E0.? = Beispiele)
OUT00=E0.0
OUT01=E0.1
OUT02=E0.2
...
Items zum Empfang von Daten (A0.? = Beispiele)
IN00=A0.0
IN01=A0.1
IN02=A0.2
...

Die Definition von IN und OUT muss bei 00 (Null nicht Buchstabe O) beginnen und fortlaufend sein. Die zu sendenden Daten müssen den Common-Integer-Variablen CI100 und nachfolgend zugewiesen werden. Auf die empfangenen Daten kann über die Common-Integer-Variable CI200 und nachfolgend zugegriffen werden.
Die Gültigkeit des DDE-Kanals wird im Statusfenster von McuWIN angezeigt.
Die DDE-Kanäle werden derzeit mit einer Taktrate von 20ms bedient.

13 Zusatzprogramme

13.1 RegDisp.EXE

Mit diesem Tool können Register (CI-Variable), digitale Ein- und Ausgänge und PCI-Ressourcen bitweise angezeigt und editiert werden. Hierbei können die einzelnen Werte seitenweise gruppiert und beschriftet werden. Somit ist es möglich, Zustandsinformationen der Steuerung bzw. der Anlage übersichtlich darzustellen und manuelle Eingriffe vorzunehmen. Dieses Tool ist insbesondere bei der Programmentwicklung und Inbetriebnahme einer Anlage hilfreich.
Die Konfiguration der Seiten und Elemente erfolgt jeweils durch eine Menüauswahl, die bei Klick mit der rechten Maustaste auf die jeweiligen Elemente geöffnet wird. Die so programmierten Elemente werden in der Konfigurationsdatei Inicfg.ini gespeichert. Nach jedem Neustart wird der vorher erstellte Zustand wiederhergestellt. Will man die gesamte Konfiguration verwerfen, so muss einfach die Konfigurationsdatei IniCfg.ini entfernt werden.

13.2 ToolEdit.EXE

Mit diesem Programm können die Einträge in der Werkzeug-Korrekturtabelle (ToolComp.ini) editiert werden. Die Felder Hor.Feed und Vert.Feed sind für zukünftige Erweiterungen vorgesehen und haben derzeit keine Funktion. Ab Version 2.5.3.2 werden beim Speichern die aktuellen Werte sofort auf die Steuerung übertragen.
Für jede Hauptebene kann eine eigene Werkzeugtabelle erfasst werden. Bei der Ebenenauswahl "All Planes" werden Werkzeuge erfasst, welche für alle Ebenen gültig sind. Bei Werkzeugnummern, welche sowohl unter "All Planes" als auch bei einer Hauptebene erfasst sind, wird für die jeweilige Ebene der dafür spezifizierte Wert verwendet.
Bei jedem Start von McuWIN wird die Werkzeugtabelle aus ToolComp.ini ausgelesen und auf die Steuerungsbaugruppe übertragen.

14 Hinweis zur Verwendung mit der PA 8000 und PS840

Für die ISA-Karten PA 8000 und PS840 existiert auch eine Programmversion von McuWIN. Bei dieser Variante müssen jedoch beim Einrichten und bei der Verwendung einige Besonderheiten bzw. Einschränkungen beachtet werden.

14.1 Besonderheiten bei der Installation

Für die Installation kann nicht das gleiche Setup verwendet werden wie bei den PCI-Karten. Es ist ein gesondertes Installationspaket erforderlich. Nach der Installation wird zunächst die Kommunikation mit dem Programm mcfg überprüft und die Basisadresse der Karte eingestellt. Diese Basisadresse muss in der Konfigurationsdatei McuWIN.ini manuell eingetragen werden, falls der Wert 300hex nicht eingestellt ist.

Eintragung in der Section [MCU] bei der Variablen BaseAdress:
768 entspricht 300 hex

Falls ein Hexadezimalwert eingetragen werden soll, kann dies mit vorangestelltem 0x erfolgen.
Beispiel: BaseAdress=0x320

14.2 Einschränkungen mit der PA 8000 und PS840

Die ISA-Karten PA 8000 und PS840 beinhalten nicht alle Funktionen der Karten APCI-8001, APCI-8008 und CPCI-8004. Hierzu sind die entsprechenden Handbücher zu beachten. Insbesondere die Bereiche Look-Ahead und Werkzeugradiuskorrektur stehen bei diesen Karten nicht zur Verfügung.

15 Fehlerdiagnose

- **Beim Start eines Programms wird nach dem Übersetzungsvorgang Fehler 21 angezeigt:**
  In diesem Fall ist das übersetzte Programm zu groß für den Task Arbeitsspeicher der Anwendertask (TASK 3). In diesem Fall muss die Systemvariable SZTSK3 der Steuerung auf einen geeigneten Wert gesetzt werden. Dieser Wert wird in Bytes angegeben und kann durchaus einige MB betragen. Standard ist 100000.
  Das Setzen von Umgebungsvariablen ist beschrieben im IHB Kapitel „Umgebungsvariable der Steuerungshardware“.

- **Beim Kompilieren von Task1.SRC in mcfg taucht der Fehler 89 auf:**
  Um dieses File zu übersetzen, muss die Option „Full System“ gesetzt sein.

- **Achse fährt bei Referenzfahrt zyklisch auf den Referenzschalter und wieder davon weg:**
  Als Referenzschaltereingang wird nicht der achsspezifische Eingang für Positionslatch verwendet (siehe Abschnitt Hardware-Voraussetzungen / Referenzschalter). Der Referenzschalter der entsprechenden Achsen muss anders verdrahte, oder die Referenzfahrtroutine in TASK0.SRC muss abgeändert werden.

- **Task 0 bzw. Task 1 lassen sich nicht übersetzen:**
  In der ersten Quelltextzeile erscheint die Fehlermeldung: Error 4, Duplicate Identifier
  Ursache dafür ist, dass ein reservierter Systemname als Achsname verwendet wird. Überprüfen aller Achsnamen.

- **Während der Referenzierung tritt der Fehler „Konflikt in den Konfigurationsdaten!“ (20hex) auf:**
  Ursache hierfür kann sein:
  - mindestens ein Wert der JOG- Geschwindigkeiten oder Beschleunigungen ist mit 0 initialisiert (jvl, jac, hvl. hac)
  - im Ini-File ist in der Section [REFERENZFAHRT] unter dem Wert Reihenfolge eine Achse mehrfach angegeben
  - im Ini-File ist in der Section [REFERENZFAHRT] unter dem Wert Reihenfolge eine Achse angegeben, die entweder auf der APCI-8001 nicht verfügbar ist, oder die im File GCODE.INC in der Konstante BitAll nicht angegeben ist.

- **Andere Ursachen für den Fehler „Konflikt in den Konfigurationsdaten!“ (20hex):**
  - in GCode.INC sind die Achszuordnungen nicht in Ordnung
    (NrXAxis, NrYAxis, NrZAxis)

- **Status # 1000: Laufzeitfehler in SAP Task!**
  Dieser Fehler zeigt einen Fehler in der Programmierung der APCI-8001 Taskumgebung an. So ist es z.B. möglich, dass eine Programmoption verwendet wird, die von der aktuellen RWMOS-Betriebssystemsoftware nicht unterstützt wird.
  Mit Hilfe folgender Vorgehensweise kann dieser Fehler lokalisiert werden:
  Aufruf von mcfg.exe und öffnen eines Fensters „Cnc Task Status“ z.B. mit Shift-F5
  inspizieren, ob bei einer Task ein Laufzeitfehler ansteht. Falls ja, kann hier die Fehlerzeile und die Programmdatei ermittelt werden.

- **Ein unbekannter Fehler oder unbekannte Warnung wird angezeigt**
  Innerhalb von Applikationserweiterungen ist es möglich, die Bitbelegung der Fehler der Warn-Variable zu ergänzen. Hierzu muss dann in McuWIN.INI ein Fehlertext definiert werden (siehe Kapitel 5.7.1 und 5.8.1).

- **Status #40: Fehler bei Referenzfahrt**
  In CI17 ist die verursachende Achse bitcodiert angezeigt. Dieser Fehler kann folgende Ursachen haben:
  - Endschalterbereich kann mit dem Freifahrweg nicht verlassen werden
  - beim Referenzieren auf einen Endschalter wird kein Endschalter gefunden
  - beim Referenzieren auf das Indexsignal wird Dieses nicht gefunden